

# A framework for multi-objective optimisation based on a new self-adaptive particle swarm optimisation algorithm

Biwei Tang<sup>1</sup>, Zhanxia Zhu<sup>1</sup>, Hyo-Sang Shin<sup>2\*</sup>, Antonios Tsourdos<sup>2</sup>, Jianjun Luo<sup>1</sup>

<sup>1</sup>*National Key Laboratory of Aerospace Flight Dynamics, School of Astronautics, Northwestern Polytechnical University, Xi'an Shaanxi 710072, China*

<sup>2</sup>*School of Aerospace, Transport and Manufacturing, Cranfield University, Bedfordshire, MK43 0AL, UK*

---

## Abstract

This paper develops a particle swarm optimisation (PSO) based framework for multi-objective optimisation (MOO). As a part of development, a new PSO method, named self-adaptive PSO (SAPSO), is first proposed. Since the convergence of SAPSO determines the quality of the obtained Pareto front, this paper analytically investigates the convergence of SAPSO and provides a parameter selection principle that guarantees the convergence. Leveraging the proposed SAPSO, this paper then designs a SAPSO-based MOO framework, named SAMOPSO. To gain a well-distributed Pareto front, we also design an external repository that keeps the non-dominated solutions. Next, a circular sorting method, which is integrated with the elitist-preserving approach, is designed to update the external repository in the developed MOO framework. The performance of the SAMOPSO framework is validated through 12 benchmark test functions and a real-world MOO problem. For rigorous validation, the performance of the proposed framework is compared with those of four well-known MOO algorithms. The simulation results confirm that the proposed SAMOPSO outperforms its contenders with respect to the quality of the Pareto front over the majority of the studied cases. The non-parametric comparison results reveal that the proposed method is significantly better than the four algorithms compared at the confidence level of 90% over the 12 test functions.

**Keywords:** Multi-objective optimisation, Self-adaptive particle swarm optimisation, Convergence of particle swarm optimisation, Circular sorting method.

---

## 1. Introduction

Over the last few decades, the multiple-objective optimisation (MOO) has gained great attentions in different areas such as manufacturing optimisation [1, 2] and environmental/economic dispatch [3]. Since there may exist conflicts among different objectives in a MOO problem, it is difficult or even impossible to simultaneously optimise all objectives in a MOO problem [4, 5, 6]. Therefore, the research of MOO often leads to a problem finding a set of non-dominated solutions [4, 7, 8]. The issue is that many real-world MOO problems may contain multiple complex and nonlinear objectives and constraints [4, 5, 6]. With the complexity and nonlinearity of objectives and constraints, finding a set of good quality non-dominated solutions become more challenging [4, 5, 6].

Thanks to their population-based nature and inherent parallelism, various evolutionary algorithms (EAs) have been proposed for solving different MOO problems. For instances, a novel gradient-based water cycle algorithm (GWCA) with evaporation rate was developed by Alireza et. al. in [9]; an adaptive gradient descent-based local search in memetic algorithm was presented to handle the optimal controller design problem by Aliasghar and Alireza in [10]; Li and Zhang developed a new version of multi-objective evolutionary method based on the differential evolution algorithm (MOEA/D-DE) to solve MOO problems with complicated Pareto sets in [11] and a multi-objective memetic

---

\*Corresponding author

Email addresses: b.tang@cranfield.ac.uk (Biwei Tang<sup>1</sup>), zhuzhanxia@sina.com (Zhanxia Zhu<sup>1</sup>), h.shin@cranfield.ac.uk (Hyo-Sang Shin<sup>2</sup>), a.tsourdos@cranfield.ac.uk (Antonios Tsourdos<sup>2</sup>), 2271304917@qq.com (Jianjun Luo<sup>1</sup>)

algorithm based on decomposition (MOEA/D) was proposed by Tan et. al. to solve different MOO problems in [12]. Some other excellent works that concentrate on applying different EAs to tackle with different MOO problems can be found in [3, 6, 13, 14].

As one of the most well-known and preferred EAs, particle swarm optimisation (PSO) has been rapidly and widely applied to solve different single-objective and MOO problems in recent years: a novel adaptive particle swarm optimisation (APSO) algorithm was developed by Alireza and Hamidreza in [15]; Yashar and Alireza proposed a novel fractional PSO-based memetic algorithm (FPSOMA) to solve trajectory control in [16]; a bare-bones multi-objective PSO for the environmental/economic dispatch problem was developed in [17]; a modified binary PSO-based reliability redundancy allocation method was introduced in [18] and a hybrid PSO-based MOO method was proposed to handle the flexible job-shop scheduling problem in [19]. For more works focusing on developing different PSO-based MOO methods, the reader can be referred to [5, 20, 21, 22].

Nevertheless, since the basic PSO algorithm cannot well balance exploration and exploitation [23, 24], the Pareto front searched by the basic PSO may converge to a false Pareto front [5]. This could limit the application of PSO on MOO. It is of great importance to overcome this convergence issue to improve the quality of the Pareto front and consequently enhance the performance of MOO [5, 17]. There have been numerous researches focusing on overcoming the typical drawback of the basic PSO [9, 23, 24, 25, 16, 26]. From these studies, it is clearly evident that adjusting the three control parameters, i.e., the inertial weight, the cognitive and social acceleration parameters, is a powerful remedy to the convergence issue in PSO. The three control parameters of PSO influence its exploration and exploitation capabilities and thus determine its convergence property. Therefore, it is essential to address and guarantee the convergence of PSO when adjusting the three parameters for improving PSO [21, 27, 28]. However, like in most of the stochastic approaches, the stochastic nature of PSO imposes difficulties on the analytical investigation of its convergence [29].

This paper first develops a novel self-adaptive PSO algorithm, called self-adaptive PSO (SAPSO). The main focus of the development is to alleviate the convergence issue of the basic PSO through fine-tuning the three main control parameters. The new self-adaptive strategy proposed adjusts the three control parameters of particles in SAPSO to well balance the trade-offs between exploration and exploitation. In the proposed self-adaptive strategy, the search of particles leverages not only the relative importance between exploration and exploitation over iterations, but also the information of the solution space. As discussed, since the convergence of PSO is a paramount issue in the context of MOO, this paper theoretically analyses the convergence of SAPSO and proposes a parameter selection principle, guaranteeing the convergence of SAPSO.

Then, this paper develops a MOO framework, named self-adaptive multi-objective PSO (SAMOPSO), based on the SAPSO algorithm proposed. Similar to the most currently existing PSO-based MOO approaches, an external repository is designed in the MOO framework to save the non-dominated personal best solutions of particles. For obtaining a well-distributed Pareto front, we introduce a circular sorting method, which is combined with the elitist-preserving approach [4] and updates the external repository.

The performance of the proposed approach is validated through 12 well-known MOO benchmark test functions and a real-world engineering problem. For rigorous verification, the performance of the proposed approach is compared with those of four well-established MOO approaches, namely NSGA-II [4], TV-MOPSO [18], BMOPSO [30] and MOEA/D [31]. The performance comparison is based on five different MOO performance metrics. The simulation results confirm that the proposed approach outperforms its contenders in terms of the quality of the obtained Pareto fronts over the majority of the cases studied. The analysis results of non-parametric statistical comparison also verify that the proposed method performs significantly better than the other four MOO methods with the confidence level of 90% for the 12 benchmark test functions. Furthermore, the computation time of the proposed approach is comparable with those of its counterparts in all the test problems.

The remainder of this paper is organised as follows. Section 2 introduces the proposed SAPSO and investigates its convergence properties. The proposed SAPSO-based MOO framework is described in Section 3. Section 4 performs the numerical simulations and discusses its results. Conclusions of this study are provided in Section 5.

## 2. Particle swarm optimisation (PSO)

### 2.1. Review of the basic PSO

Inspired by birds flocking and fish schooling, Kennedy and Eberhart first proposed PSO in 1995. The original aim of the basic PSO algorithm is to reproduce the social interactions among agents to solve some complex optimisation problems [29]. Each agent in PSO is called a particle and associated with a velocity, which is dynamically adjusted depending on its own flight experience and those of its companions. Therefore, each particle is attracted toward a stochastically weighted average of its personal best position and the global best position of the swarm. In the basic PSO algorithm, from iteration  $k$  to  $k + 1$ , each particle updates its velocity and position as follows:

$$V_m^{k+1} = wV_m^k + c_1r_1(pbest_m^k - X_m^k) + c_2r_2(gbest - X_m^k) \quad (1)$$

$$X_m^{k+1} = X_m^k + V_m^{k+1} \quad (2)$$

where  $w$  is a real coefficient denoting the inertia weight.  $c_1$  and  $c_2$  are two positive real coefficients representing the cognitive and social acceleration parameters, respectively.  $r_1$  and  $r_2$  are two random numbers uniformly distributed in  $[0, 1]$ .  $pbest_m^k$  denotes the personal best position of the  $m$ th particle at iteration  $k$ .  $gbest$  denotes the global best position of the swarm.

### 2.2. The proposed SAPSO

When using PSO to solve an optimisation problem, it is essential to properly control the exploration and exploitation capabilities of particles to efficiently find the global optimum [23, 24, 25]. Ideally, on one hand, the exploration capability needs to be facilitated in the early stage of the evolution, so that particles can wander through the entire solution space, rather than clustering around the current population-best solution [23, 24, 25]. On the other hand, the exploitation capability is required to be promoted in the late stage of the evolution, so that particles can focus on the local search to increase the possibility of finding optimal solutions [23, 24, 25].

It is well known that the exploration and exploitation capabilities of PSO heavily depend on the three control parameters of particles. The basic philosophies concerning how the three control parameters influence such two abilities of PSO can be summarised as follows: (1) a large inertia weight enhances exploration, while a small inertia weight facilitates exploitation [23, 24, 25]; (2) a large cognitive component, compared with the social component, results in wandering of particles through the entire search space, which thus strengthens exploration [23, 24, 25]; (3) a large social component, compared with the cognitive component, leads particles to a local search, which consequently intensifies the exploitation capability [23, 24, 25].

Although MOO generally aims at obtaining a set of non-dominated solutions rather than a single optimal solution, how to achieve a good balance between the exploration and exploitation capabilities of PSO also remains a key issue in finding better non-dominated solutions [5, 32]. Focusing on enhancing the performance of PSO, this paper develops a new PSO algorithm, called SAPSO. The main purpose of the development of SAPSO is to improve the performance of SAPSO by tuning its three control parameters in a way well balancing the exploration and exploitation capabilities of particles. The proposed SAPSO is then integrated into MOO to find high-quality non-dominated solutions. For fine-tuning the three control parameters, we propose a novel self-adaptive strategy as:

$$w_m^{k+1} = (w_{max} - w_{min}) \exp(-\frac{\delta_w k}{\beta_m}) + w_{min} \quad (3)$$

$$c_{1m}^{k+1} = (c_{1s} - c_{1f}) \exp(-\frac{\delta_{c1} k}{\beta_m}) + c_{1f} \quad (4)$$

$$c_{2m}^{k+1} = (c_{2s} - c_{2f}) \exp(\frac{\delta_{c2} k}{\beta_m}) + c_{2f} \quad (5)$$

$$\delta_w = \frac{w_{max} - w_{min}}{k_{max}} \quad (6)$$

$$\delta_{c1} = \frac{c_{1s} - c_{1f}}{k_{max}} \quad (7)$$

$$\delta_{c2} = \frac{c_{2s} - c_{2f}}{k_{max}} \quad (8)$$

$$\beta_m = \frac{\|V_m^{k-1}\| + \|V_m^k\|}{2\|V_m^{k-1}\| + \Delta} \quad (9)$$

where  $w_{max}$  and  $w_{min}$  are the upper and lower bounds of the inertia weight, respectively.  $c_{1s}$  and  $c_{1f}$  are the initial and final values of the cognitive acceleration parameter.  $c_{2s}$  and  $c_{2f}$  denote the initial and final values of the social acceleration parameter.  $k_{max}$  denotes the maximum iteration number.  $\|V_m^{k-1}\|$  and  $\|V_m^k\|$  represent the  $L_2$ -norm of the velocity vector of the  $m$ th particle at iterations  $(k-1)$  and  $k$ , respectively.  $\Delta$  is a sufficiently small positive real number ( $\Delta = 1e-25$  in this paper). Note that  $c_{1s} > c_{1f}$  and  $c_{2s} < c_{2f}$  in the self-adaptive strategy proposed.

### 2.3. Parametric analysis for SAPSO

From Eqns. (3)-(5), it clear that  $w$  and  $c_1$  decrease ( $c_2$  increases) as the iteration number  $k$  increases. Therefore, based on the aforementioned basic philosophies, SAPSO is likely to start the search with a high exploration tendency in the early stage of the evolution. As the two parameters decrease over time, the exploitation capability of SAPSO becomes favored in the late stage of the evolution. Note that, following the updating rule for a fixed  $\beta_m$ , the balance between exploration and exploitation in SAPSO varies only with respect to the iteration number  $k$ .

Apart from the iteration number  $k$ , the balance of the search in SAPSO is also adapted by the additional parameter  $\beta_m$ . From Eqns. (3)-(4), it is trivial that the changes in  $w$  and  $c_1$  become smaller as  $\beta_m$  becomes larger. On the other hand, it is clear from Eq. (5) that the variation in  $c_2$  becomes larger as the  $\beta_m$  value increases. This implies that, for a large value of  $\beta_m$ , the exploration capability tends to be more dominant. In contrast, the exploitation capability takes over the exploration ability more quickly as  $\beta_m$  decreases.

Eqn. (9) indicates that  $\beta_m$  increases as  $\|V_m^k\|$  becomes relatively larger than  $\|V_m^{k-1}\|$ , i.e., the difference between the two consecutive positions of particle  $m$  gets bigger. In such a case where the distance between the two positions of the particle becomes larger, it is logical to keep the exploration capability in order not to miss potentially important solution space during the search. In the case where  $\beta_m$  is relatively smaller, it is also desirable to promote the exploitation capability of SAPSO to increase the possibility of finding optimal solutions. This exactly complies with the tendency of changes in the three control parameters of the proposed self-adaptive strategy in SAPSO.

By utilising the proposed self-adaptive strategy, the three control parameters of particles in SAPSO can be adjusted in a way complying with the basic philosophies of the PSO development. Consequently, the proposed SAPSO is expected to improve the ability in finding high-quality solutions. Fig. 1 demonstrates the tendency of these changes in the three control parameters with respect to different values of  $\beta_m$ . Note that  $w_{max} = 0.9$ ,  $w_{min} = 0.1$ ,  $c_{1s} = c_{2f} = 2.5$ ,  $c_{1f} = c_{2s} = 0.5$  and  $k_{max} = 100$  in this demonstration.

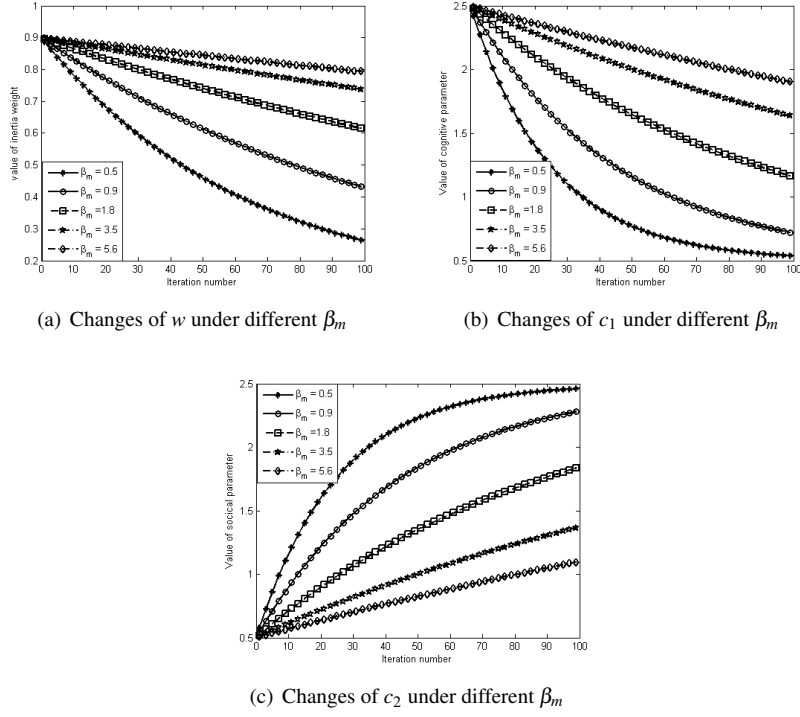


Fig. 1. Changes of the three control parameters under different  $\beta_m$  in SAPSO

#### 2.4. Analysis of trajectory in SAPSO

This subsection first analytically investigates the convergence of SAPSO. Then, a parameter selection principle guaranteeing the convergence of SAPSO is proposed.

##### 2.4.1. Convergence analysis of SAPSO

Since each dimension of velocity and position vectors of each particle is updated independently from the others in Eq. (1) and Eq. (2), SAPSO can be simplified into a one-dimensional case for the analysis of its convergence property. For simplicity, we omit the subscript  $m$  in Eq. (1) and Eq. (2). Then, the moving rules of particles in the one-dimensional SAPSO can be rewritten into a matrix form:

$$\begin{bmatrix} X(k+1) \\ V(k+1) \end{bmatrix} = \mathbf{A} \begin{bmatrix} X(k) \\ V(k) \end{bmatrix} + \mathbf{B}P \quad (10)$$

where

$$\mathbf{A} = \begin{bmatrix} 1 - \phi & w \\ -\phi & w \end{bmatrix} \quad (11)$$

$$\mathbf{B} = [\phi \ \phi]^T \quad (12)$$

$$\phi = \phi_1 + \phi_2 \quad (13)$$

$$\phi_1 = c_1 r_1 \quad (14)$$

$$\phi_2 = c_2 r_2 \quad (15)$$

$$P = \frac{\phi_1 \cdot pbest + \phi_2 \cdot gbest}{\phi_1 + \phi_2} \quad (16)$$

Solving  $|\lambda \mathbf{E} - \mathbf{A}| = 0$ , where  $\mathbf{E}$  is the identity matrix with the same size of  $\mathbf{A}$ , the characteristic equation to the system Eq. (10) is derived as:

$$\lambda^2 - (w + 1 - \phi)\lambda + w = 0 \quad (17)$$

where two roots, denoted by  $\lambda_{1,2}$ , are obtained as:

$$\lambda_{1,2} = \frac{1 + w - \phi \pm \sqrt{(1 + w - \phi)^2 - 4w}}{2} \quad (18)$$

<sup>145</sup> In the context of the dynamic system theory, the necessary and sufficient condition for the convergence of the system represented by Eq. (10) is that magnitudes of  $\lambda_{1,2}$  are less than 1 [33]. Thus, the system given by Eq. (10) converges if and only if:

$$\text{Max}\{|\lambda_1|, |\lambda_2|\} < 1 \quad (19)$$

From Eq. (18), it is clear that  $\lambda_{1,2}$  are either two real numbers or complex conjugates. Therefore, we investigate the convergence property in the two cases.

<sup>150</sup> **Case 1.**  $\lambda_{1,2}$  are complex numbers, i.e.  $\lambda_{1,2} \in \mathbb{C}$ .

**Lemma 1.** *In the system represented by Eq. (10),  $\lambda_{1,2} \in \mathbb{C}$  if and only if:*

$$\begin{cases} w - 2\sqrt{w} + 1 < \phi < w + 2\sqrt{w} + 1 \\ w \geq 0 \end{cases} \quad (20)$$

*Proof.* From Eq. (10), it is clear that:

$$\lambda_{1,2} \in \mathbb{C} \Leftrightarrow (1 + w - \phi)^2 - 4w < 0 \quad (21)$$

Solving (21) by the classical mathematical approach, Lemma 1 can be easily proved.  $\square$

<sup>155</sup> Now, let us find conditions on  $\phi$  and  $w$  guaranteeing the convergence of the system given by Eq. (10) for  $\lambda_{1,2} \in \mathbb{C}$ . It is trivial that the system Eq. (10) converges if and only if  $\text{Max}\{|\lambda_1|, |\lambda_2|\} < 1$ .

**Lemma 2.** *For  $\lambda_{1,2} \in \mathbb{C}$ , the system given by Eq. (10) converges, if and only if:*

$$\begin{cases} w - 2\sqrt{w} + 1 < \phi < w + 2\sqrt{w} + 1 \\ 0 \leq w < 1 \end{cases} \quad (22)$$

*Proof.* Note that the absolute value of a complex number  $Z$  can be computed as  $|Z| = \sqrt{Z_r^2 + Z_c^2}$ , where  $Z_r$  and  $Z_c$  denote the real and imaginary parts of  $Z$ . Hence, for  $\lambda_{1,2} \in \mathbb{C}$ , we have:

$$\text{Max}\{|\lambda_1|, |\lambda_2|\} = |\lambda_1| = |\lambda_2| = \sqrt{w} \quad (23)$$

Therefore:

$$\text{Max}\{|\lambda_1|, |\lambda_2|\} < 1 \Leftrightarrow \sqrt{w} < 1 \quad (24)$$

<sup>160</sup> For  $\lambda_{1,2} \in \mathbb{C}$ , Eq. (20) must hold according to **Lemma 1**. From the two conditions, i.e. Eq. (20) and Eq. (24), the system represented by Eq. (10) converges, if and only if:

$$\begin{cases} w - 2\sqrt{w} + 1 < \phi < w + 2\sqrt{w} + 1 \\ 0 \leq w < 1 \end{cases} \quad (25)$$

$\square$

**Case 2.**  $\lambda_{1,2}$  are two real numbers, i.e.  $\lambda_{1,2} \in \mathbb{R}$ .

**Lemma 3.** For the system given by Eq. (10),  $\lambda_{1,2} \in \mathbb{R}$  if and only if:

$$\begin{cases} \phi \in \mathbb{R} & \text{for } w < 0 \\ \phi \leq w - 2\sqrt{w} + 1 \text{ or } \phi \geq w + 2\sqrt{w} + 1 & \text{for } w \geq 0 \end{cases} \quad (26)$$

<sup>165</sup> *Proof.* For the system in Eq. (10), it is clear that:

$$\lambda_{1,2} \in \mathbb{R} \Leftrightarrow (1 + w - \phi)^2 - 4w \geq 0 \quad (27)$$

Solving the right-hand side of Eq. (27) using the classical approach completes the proof.  $\square$

Let us now investigate  $\phi$  and  $w$  conditions for the convergence guarantee in Case 2. From Eq. (18) and Eq. (19), it is trivial, for  $\lambda_{1,2} \in \mathbb{R}$ , that the condition  $\text{Max}\{|\lambda_1|, |\lambda_2|\} < 1$  holds if and only if:

$$-1 < \frac{1 + w - \phi \pm \sqrt{(1 + w - \phi)^2 - 4w}}{2} < 1 \quad (28)$$

Hence:

$$-3 - w + \phi < \pm \sqrt{(1 + w - \phi)^2 - 4w} < 1 - w + \phi \quad (29)$$

<sup>170</sup> For  $\lambda_{1,2} \in \mathbb{R}$ , it is obvious that:

$$(28) \Leftrightarrow \begin{cases} -3 - w + \phi < -\sqrt{(1 + w - \phi)^2 - 4w} \\ \sqrt{(1 + w - \phi)^2 - 4w} < 1 - w + \phi \end{cases} \quad (30)$$

Solving the right-hand side inequalities in Eq. (30), we have:

$$(28) \Leftrightarrow \begin{cases} 2w + 2 - \phi > 0 \\ \phi > 0 \end{cases} \quad (31)$$

Therefore, it is clear that, for  $\lambda_{1,2} \in \mathbb{R}$ , the system in Eq. (10) converges if and only if:

$$\begin{cases} 0 < \phi < 2w + 2, -1 < w < 0 \\ 0 < \phi \leq w - 2\sqrt{w} + 1 \text{ or } w + 2\sqrt{w} + 1 \leq \phi < 2w + 2, 0 \leq w < 1 \end{cases} \quad (32)$$

Considering Case 1 and Case 2 together, the system represented by Eq. (10) converges if and only if:

$$\begin{cases} 0 < \phi < 2w + 2 \\ -1 < w < 1 \end{cases} \quad (33)$$

where  $\phi = \phi_1 + \phi_2 = c_1 r_1 + c_2 r_2$ .

<sup>175</sup> It is important to notice that the condition given by Eq. (33) is the necessary and sufficient condition for the convergence of SAPSO. The convergence region denoted by Eq. (33) is shown in Fig. 2. For the parameter selection that locates  $w$  and  $\phi$  in the convergence region given in Eq. (3), the trajectory convergence can be guaranteed in SAPSO.

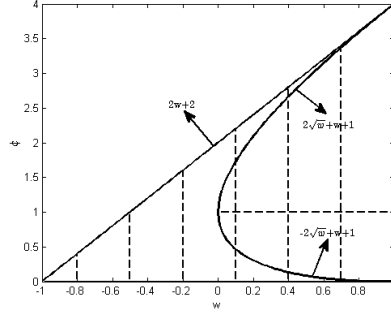


Fig. 2. The convergence region for SAPSO

#### 2.4.2. The equilibrium point of SAPSO

In the previous subsection, the convergence property of SAPSO is analytically investigated. Now, the remaining task is to find the equilibrium point of SAPSO. Note that the equilibrium point is a stable point, to which particles converge. Calculating limits on both sides of Eq. (10) gives:

$$\begin{cases} \lim_{k \rightarrow \infty} X(k+1) = w \lim_{k \rightarrow \infty} V(k) + \phi \lim_{k \rightarrow \infty} (P - X(k)) \\ \lim_{k \rightarrow \infty} V(k+1) = \lim_{k \rightarrow \infty} X(k) + \lim_{k \rightarrow \infty} V(k) \end{cases} \quad (34)$$

When SAPSO converges, it is clear that  $\lim_{k \rightarrow \infty} X(k+1) = \lim_{k \rightarrow \infty} X(k)$  and  $\lim_{k \rightarrow \infty} V(k+1) = \lim_{k \rightarrow \infty} V(k)$ . Thus, substituting these two equations into Eq. (34), yields:

$$\begin{cases} \lim_{k \rightarrow \infty} X(k) = P = \frac{\phi_1 \cdot pbest + \phi_2 \cdot gbest}{\phi_1 + \phi_2} \\ \lim_{k \rightarrow \infty} V(k) = 0 \end{cases} \quad (35)$$

where  $\phi_1 = c_1 r_1$  and  $\phi_2 = c_2 r_2$ .

#### 2.4.3. Convergence behaviour of particles in SAPSO

Before particles converge to the equilibrium point given in Eq. (35), they may exhibit different convergence behaviours depending on values of  $w$  and  $\phi$ . Four typical convergence behaviours of particles in SAPSO are shown in Fig. 3.

A non-oscillatory behaviour as shown in Fig. 3(a) leads particles to search only on one side of the equilibrium point. Particles exhibit the non-oscillatory convergence behaviour when  $\lambda_1$  and  $\lambda_2$  are two real roots and at least one of them is positive, which is equivalent to  $0 \leq (1+w-\phi)^2 - 4w$  and  $0 < 1+w-\phi$ . The harmonic oscillation behaviour demonstrated in Fig. 3(b) occurs in the case where  $\lambda_1$  and  $\lambda_2$  are two complex roots, i.e.  $(1+w-\phi)^2 - 4w < 0$ . Particles present zigzagging convergence behaviour shown in Fig. 3(c) when at least one of  $\lambda_1$  and  $\lambda_2$  has a negative real part, which is equal to  $w < 0$  or  $1+w-\phi < 0$ . The combined harmonic with zigzagging behaviour illustrated in Fig. 3(d) combines the harmonic and the zigzagging behaviours, which thus emerges when at least two complex  $\lambda_1$  and  $\lambda_2$  roots has a negative real part, that is  $(1+w-\phi)^2 - 4w < 0 \cap w < 0 \cup (1+w-\phi)^2 - 4w < 0 \cap 1+w-\phi < 0$ .

If boundaries of coefficients associated with these convergence behaviours are known beforehand, one may easily design an adaptive method to change values of these coefficients, so that the convergence of PSO can be guaranteed and the quality of the final solution can be improved.



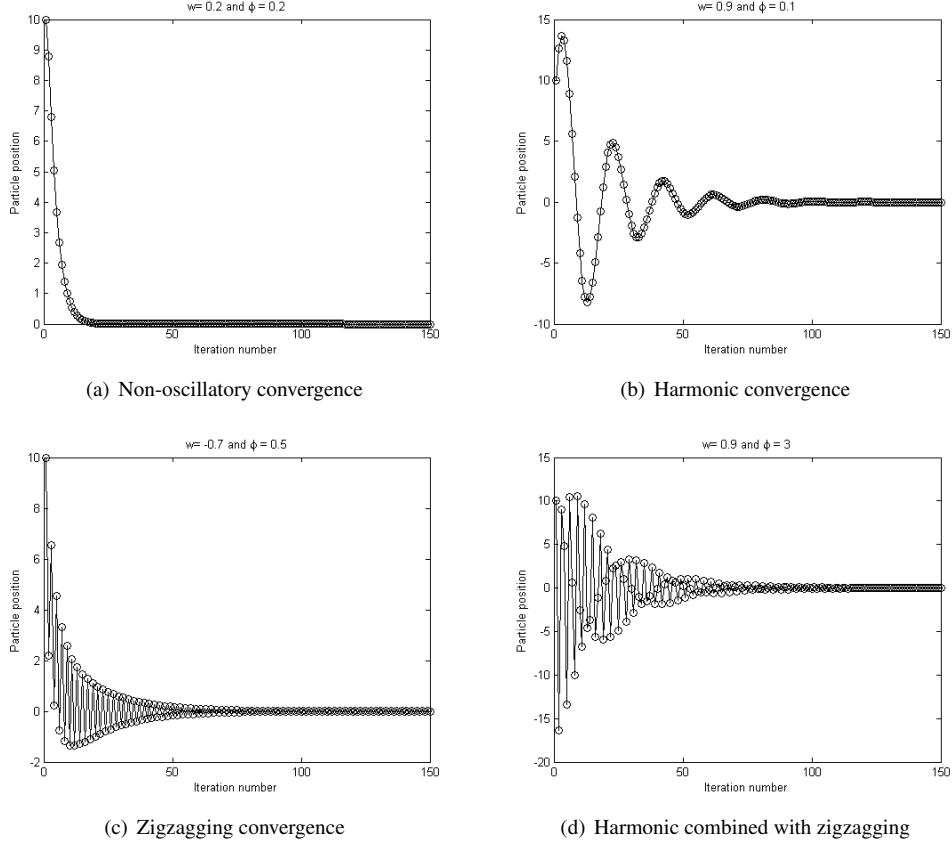


Fig. 3. Convergence behaviour of particles in SAPSO

#### 2.4.4. Convergence property of SAPSO with its stochastic nature

Due to its stochastic nature, it is difficult to rigorously establish an exact relationship between the stochastic nature and the convergence of SAPSO. However, one can still analyse the convergence property of SAPSO considering the bound of stochastic numbers used in SAPSO. Note that the stochastic nature of SAPSO is attributed to the existence of two random numbers  $r_1$  and  $r_2$ .

Since  $\phi = \phi_1 + \phi_2 = c_1 r_1 + c_2 r_2$ , the necessary and sufficient condition given by Eq. (33) can be rewritten as:

$$\begin{cases} 0 < c_1 r_1 + c_2 r_2 < 2w + 2 \\ -1 < w < 1 \end{cases} \quad (36)$$

**Lemma 4.** SAPSO converges if:

$$\begin{cases} 2w + 2 > c_1 + c_2 \\ -1 < w < 1 \\ c_1, c_2 > 0 \end{cases} \quad (37)$$

*Proof.* Because  $c_1$  and  $c_2$  are two positive parameters, and  $r_1$  and  $r_2$  are two random numbers uniformly distributed in  $[0, 1]$ , it is trivial that  $c_1 \geq c_1 r_1$  and  $c_2 \geq c_2 r_2$ . Therefore:

$$\begin{cases} 2w + 2 > c_1 + c_2 \\ -1 < w < 1 \\ c_1, c_2 > 0 \end{cases} \Rightarrow \begin{cases} 0 < r_1 c_1 + r_2 c_2 < 2w + 2 \\ -1 < w < 1 \end{cases} \quad (38)$$

Since the right-hand side inequality in Eq. (38) is the necessary and sufficient condition for the convergence of

SAPSO, it is trivial that Lemma 4 holds.  $\square$

#### 2.4.5. Parameter selection principle in SAPSO

The following lemma provides a parameter selection principle, i.e. how to set the initial and final values of  $w$ ,  $c_1$  and  $c_2$ , to guarantee the convergence in SAPSO.

**Lemma 5.** *SAPSO converges if the initial and final values of the three control parameters of each particle satisfy the following:*

$$\begin{cases} 2w_{min} + 2 > c_{1s} + c_{1f} \\ 1 > w_{max} > w_{min} > -1 \\ c_{1s} = c_{2f} > c_{1f} = c_{2s} > 0 \end{cases} \quad (39)$$

*Proof.* When  $c_{1s} = c_{2f}$  and  $c_{1f} = c_{2s}$ , it is clear from Eqs. (4)-(7) that  $c_1 + c_2 = c_{1s} + c_{1f}$  for any particle at any iteration in SAPSO. From Eqs. (3)-(5), it is trivial that  $w_{min} \leq w \leq w_{max}$ ,  $c_{1f} \leq c_1 \leq c_{1s}$  and  $c_{2s} \leq c_2 \leq c_{2f}$  for any particle at any iteration in SAPSO. Therefore:

$$\begin{cases} 2w_{min} + 2 > c_{1s} + c_{1f} \\ 1 > w_{max} > w_{min} > -1 \\ c_{1s} = c_{2f} > c_{1f} = c_{2s} > 0 \end{cases} \Rightarrow \begin{cases} 2w + 2 > c_1 + c_2 \\ -1 < w < 1 \\ c_1, c_2 > 0 \end{cases} \quad (40)$$

The right-hand side inequality in Eq. (40) is the sufficient condition for the convergence in SAPSO.  $\square$

Lemma 5 implies that the convergence condition in SAPSO can be readily satisfied by setting the initial and final values of the three control parameters. Fig. 4 shows the convergence trajectories of the position and velocity of the particle under the suggested parameter selection:  $w_{max} = 0.9$ ,  $w_{min} = 0.1$ ,  $c_{1s} = c_{2f} = 2$  and  $c_{1f} = c_{2s} = 0.1$ .

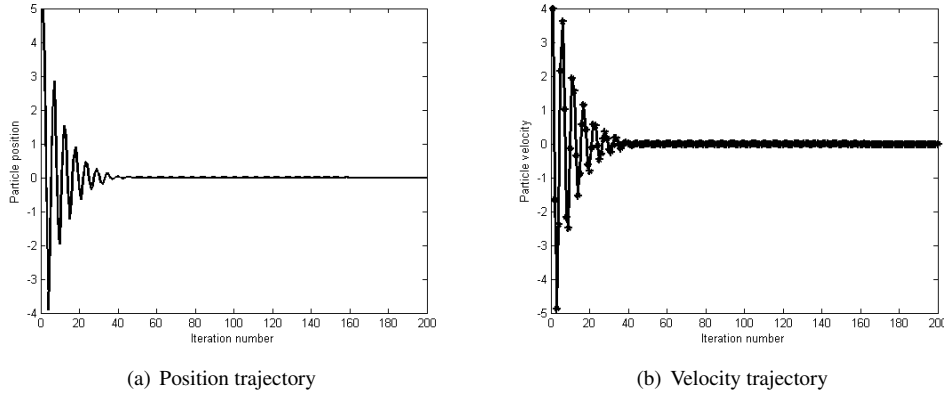


Fig. 4. Convergence position and velocity trajectories of the particle in SAPSO under the suggested parameter selection

### 3. SAPSO-based MOO framework

#### 3.1. The generality and some basic concepts of MOO

Generally, a MOO problem is to optimise a set of objectives subjecting to some equality/inequality constraints, which can be mathematically described as follows:

$$\text{Optimize : } F(x) = [f_1(x), f_2(x), \dots, f_Q(x)] \quad (41)$$

$$\text{Subject to : } \begin{cases} h_i(x) = 0 & i = 1, 2, \dots, CN_1 \end{cases} \quad (42)$$

$$\begin{cases} g_j(x) < 0 & j = 1, 2, \dots, CN_2 \end{cases} \quad (43)$$

$$\begin{cases} x_k^l \leq x_k \leq x_k^u & k = 1, 2, \dots, n \end{cases} \quad (44)$$

where  $f_d$  ( $1 \leq d \in \mathbb{N} \leq Q$ ) denotes the  $d$ th objective.  $Q$  is the total number of objectives and  $x = [x_1, x_2, \dots, x_n]$  is decision variable vector.  $n$  is the total number of decision variables.  $h_i(x)$  denotes the  $i$ th equality constraint and  $CN_1$  is the number of equality constraints.  $g_j(x)$  represents the  $j$ th inequality constraint and  $CN_2$  denotes the number of inequality constraints.  $x_k^l$  and  $x_k^u$  represent the lower and upper bounds of the  $k$ th decision variable, respectively.

For the completeness, some basic concepts of MOO stated in [6] are reproduced, especially for the minimisation problem.

**Pareto dominance:** Solution  $S_1$  is said to dominate solution  $S_2$ , denoted as  $S_1 \succ S_2$ , if and only if  $f_d(S_1) \leq f_{d_1}(S_2)$  for all  $d \in [1, 2, \dots, Q]$  and  $\exists d \in [1, 2, \dots, Q]$  such that  $f_d(S_1) < f_d(S_2)$ .

**Non-dominated solution:** Solution  $S$  is called a non-dominated solution (also called Pareto-optimal solution) if there exists no solution that can dominate solution  $S$ .

**Pareto set:** The set of all non-dominated solutions is called the Pareto set.

**Pareto front:** The image of the Pareto set in the objective space is called the Pareto front.

### 3.2. Implementation of SAPSO for MOO

When using PSO to solve MOO problems, there are some key issues need to be addressed; (1) how to handle constraints of the MOO problem; (2) how to update the global and personal best solutions of particles; (3) how to keep the non-dominated solutions found by particles; (4) how to update the obtained non-dominated solution set. This subsection first addresses these mentioned key issues. Then, the proposed SAPSO-based MOO framework i.e., SAMOPSO, is described at the end of this subsection.

#### 3.2.1. Handling constraints of MOO

Thanks to its simplicity and popularity, the constraint handling approach proposed in [34] is applied to handle constraints of the MOO problems. In this approach, the constraint violation degree of the  $m$ th solution is calculated as [34]:

$$viol_m = \sum_{i=1}^{CN_1} |h_i(x)| + \sum_{j=1}^{CN_2} \max(0, g_j(x)) \quad (45)$$

where  $CN_1$ ,  $h_i(x)$ ,  $CN_2$  and  $g_j(x)$  have same definitions as those in Eqs. (42)-(43).  $|h_i(x)|$  denotes the magnitude of  $h_i(x)$ .

After calculating the constraint violation degree and fitness values of each solution, the constrained dominance rule described in [22] is implemented to select the non-dominated solution between any two candidate solutions in SAMOPSO. The constrained dominance rule can be summarised as follows: (1) for any two solutions with different constraint violation degrees, the solution with smaller constraint violation degree dominates the solution with larger constraint violation degree; (2) for any two solutions with the same constraint violation degree, the precise Pareto dominance relationship described in the definition of ‘‘Pareto dominance’’ in Subsection 3.1 is used to select the non-dominated solution between those two candidate solutions.

It is clear that the constrained dominance rule considers infeasible solutions since they may still contain some valuable information of the solution space [22]. This could diversify the search for the non-dominated solutions and reduce the possibility of missing important part of the solution space in the search. Consequently, it could increase the chance of finding the high-quality non-dominated solutions[22]. Note that, unless otherwise specified, the constrained dominance rule is applied to determine and select the non-dominated solution between any two candidate solutions in the developed SAMOPSO framework.

In the SAMOPSO framework proposed, the boundary constraints of each decision variable, given by Eq. (44), is handled using a saturation strategy: the decision variable  $x_k$  for all  $k \in \{1, \dots, n\}$  applies the following saturation strategy [18]:

$$x_k = \begin{cases} x_k^l & \text{if } x_k < x_k^l \\ x_k^u & \text{if } x_k > x_k^u \\ x_k & \text{otherwise} \end{cases} \quad (46)$$

### 3.3. Updating the personal and global best solutions of particles

The personal and global best solutions in PSO-based MOO approaches are non-trivial from a set of non-dominated solutions generated. In this subsection, we address how to update the personal and global best solutions in our MOO framework.

The personal best solution of a particle is referred to the best position that the particle searched so far and can be viewed as the memory of the particle. The proposed MOO framework utilises the conventional method applied in some other MOO research literature such as [5, 18, 21, 22] to update the personal best solution of each particle. In the conventional method, the personal best solution and the current solution of the particle are compared with each other at each iteration. If the personal best solution of a particle dominates the current solution of the particle, the personal best solution is then kept; otherwise, the current solution becomes the personal best solution [5, 18, 21, 22].

Most of PSO-based MOO methods designs an external repository or archive to save the non-dominated solutions found [5, 18, 21, 22]. In the framework proposed in this paper, we also design a fixed-size external repository to save the non-dominated solutions. In addition, to efficiently determine the search directions of particles, the global best solution of each particle is selected from the external repository using the geographically-based method [5].

In the geographically-based method [5], the search objective explored so far is firstly mapped into different grids. Then, the currently-found non-dominated solutions are located into a coordinate system using these grids, where each solution's coordinates are defined based on its values of objective functions. After selecting a grid based on a density estimation operator, a non-dominated solution located in this grid is randomly selected as the global best solution [5].

Note that, in the density estimation operator, the more non-dominated solutions a grid contains, the less likely the grid is selected. This implies that the areas containing less non-dominated solutions are more likely to be selected [5]. Hence, particles are encouraged to explore the less crowded solution space. Consequently, we can increase the possibility to search less explored solution spaces and the chance of finding high-quality non-dominated solutions [5]. For more details about the geographically-based method, the reader is referred to [5].

### 3.4. Designing and updating the external repository

Because multiple non-dominated solutions are produced at each iteration in the developed MOO framework, the size of the repository could quickly increase if an external repository is designed to keep all the non-dominated solutions found. This could significantly increase the computation load in updating the repository as the current solutions of particles should be compared to the all non-dominated solutions saved in the repository. To mitigate this issue, the size of the repository in the proposed SAMOPSO is fixed as in [5, 18, 21, 22]. For the repository with a fixed size, how to update the repository becomes important. Therefore, how our SAMOPSO updates the repository will be described in this subsection.

To update the external repository, we propose to combine the circular sorting approach with the elitist-preserving approach [4]. In the circular sorting method, the newly-found solution of a particle is first compared with its personal best solution. If the new solution is non-dominated by the personal solution, it is allowed to be considered as a potential element of the repository.

Note that it is unnecessary to consider all new solutions as potential elements of the repository since they might be dominated by their personal best solutions. Therefore, sorting new solutions of individual particles by comparing them with their personal best solutions can relax the computational complexity in updating the repository.

The new solution sorted is then compared with all non-dominated solutions saved in the repository. The new solution is allowed to enter the repository if: (a) it is non-dominated by all non-dominated solutions in the repository or (b) it dominates any non-dominated solution saved in the repository (in this case, the dominated solutions are removed from the repository).

If the repository is full, the elitist-preserving approach [4] is applied. The elitist-preserving approach prunes the repository to obtain a size-equal repository and a well-distributed Pareto front. The approach first sorts all non-dominated solutions kept in the repository in an ascending order based on their function values. Then, the crowding distance of each non-dominated solution is calculated according to the objective functions of each non-dominated solution and those of its neighbors. Finally,  $N_{rep}$  non-dominated solutions with largest crowding distance are kept in the repository. Note that  $N_{rep}$  denotes the predefined size of the repository. The larger crowding distance the non-dominated solutions have, the wider they are spread. Therefore, keeping the most sparsely-spread non-dominated solutions in the repository can provide not only a size-equal repository, but also a widely-distributed Pareto front [4, 6]. For more details of the elitist-preserving approach, the reader is referred to [4].

### 3.5. Algorithmic scheme of SAMOPSO

The algorithmic scheme of SAMOPSO is summarised in Table 1. In the scheme,  $NP$ ,  $k$  and  $k_{max}$  represent the size of the swarm, the current iteration number and the maximum iteration number of the evolution, respectively.

Table 1. The SAPSO-based MOO (SAMOPSO) framework

---

1:	Set the simulation parameters and randomly generate an initial swarm
2:	Calculate cost functions and constraint violation degree of each particle at the initial iteration
3:	Set each particle's initial solution of as its $pbest$ at the initial iteration
4:	Update the repository by the circular sorting method at the initial iteration
5:	<b>if</b> the repository is full <b>do</b>
6:	Prune some solutions from the repository by the elitist-preserving approach
7:	<b>end if</b>
8:	<b>while</b> $k \leq k_{max}$ <b>do</b>
9:	<b>for</b> $m = 1 : NP$ <b>do</b>
10:	Select $gbest$ for particle $m$ from the repository by the geographically-based method
11:	$V_m \leftarrow wV_m + c_1r_1(pbest_m - X_m) + c_2r_2(gbest - X_m)$ % update velocity of particle $m$
12:	$X_m \leftarrow X_m + V_m$ % update position of particle $m$
13:	Modify each dimension of $X_m$ using the saturation strategy defined by Eq. (46)
14:	Calculate cost functions of particle $m$
15:	$viol_m = \sum_{i=1}^{CN_1}  h_i(x)  + \sum_{j=1}^{CN_2} \max(0, g_j(x))$ % calculate constraint violation degree of particle $m$
16:	Update $w$ , $c_1$ and $c_2$ of particle $m$ according to Eqs. (3)-(9)
17:	<b>if</b> $X_m \succ pbest_m$ or $X_m$ is non-dominated to $pbest_m$ <b>do</b>
18:	$pbest_m = X_m$
19:	Move $pbest_m$ into the repository and update the repository by the circular sorting method
20:	<b>end if</b>
21:	<b>end for</b>
22:	<b>if</b> the repository is full <b>do</b>
23:	Prune some solutions from the repository by the elitist-preserving approach
24:	<b>end if</b>
25:	$k \leftarrow k + 1$
26:	<b>end while</b>
27:	Output non-dominated solutions kept in the repository

---

## 4. Numerical Experiments

### 4.1. Description of evaluation metrics

To allow a quantitative assessment of the performance of different MOO methods, this paper adopts five widely-accepted evaluation metrics: the number of non-dominated solutions (NNS) [5, 6, 18], error ratio (ER) [5, 6, 18], generation distance (GD) [5, 6, 18], space metric (SM) [5, 6, 18] and the computation time (CT) [5, 6, 18]. The first four indices are mainly used to evaluate the quality of the obtained Pareto front [5, 6, 18].  $NNS$  denotes the number of real non-dominated solutions. The quality of the Pareto front obtained is better with a larger  $NNS$  [5, 6, 18].  $ER$  measures the non-convergence of the Pareto front toward the real Pareto front. The smaller the  $ER$  value is, the better the convergence becomes.  $GD$  is a distance measure of the obtained non-dominated solutions from those in the real Pareto front. The quality of the obtained Pareto front improves as the  $GD$  values decreases.  $SM$  is used to measure the uniformity in the spread of the Pareto front. The smaller the  $SM$  value becomes, the better the Pareto front solutions spread.  $CT$  is the computer execution time, which, to some extents, reflects the computational complexity of the MOO algorithm tested. Note that how to calculate these metrics is detailed in [5, 6, 18].

## 4.2. Simulations on benchmark test functions

### 4.2.1. Description of benchmarks and compared MOO methods

In order to verify the proposed approach, its performance is evaluated using 12 well-known benchmark test functions extracted from [35, 6, 11, 12, 36] and one real-world engineering problem: environmental/economic dispatch (EED) problem. All benchmark test functions are described in Table 2. The performance of the proposed SAMOPSO is compared with those of NSGA-II [4], TV-MOPSO [18], BMOPSO [30] and MOEA/D [31]. For rigorous validation, a Monte-Carlo experiment with 30 runs is conducted for each test function. In addition, the non-dominated solutions of each method are obtained after 400 iterations of 100 particles in each studied case. All considered MOO methods are programmed by MATLAB 2012B software on a windows-8 personal computer with i3-2350@2.30-GHz and 2-GB RAM. For all MOO algorithms tested, the size of the external repository is set to be 100. The simulation parameters for SAMOPSO are given as:  $w_{max} = 0.9$ ,  $w_{min} = 0.1$ ,  $c_{1s} = c_{2f} = 2.0$  and  $c_{1f} = c_{2s} = 0.1$ . These values are set based on the convergence analysis results discussed in Section 2.4.5. The simulation parameters for all compared methods are extracted from their corresponding literature and summarised in Table 3.

Table 2. Benchmark test functions

Fun.	Variable dimension	Variable bounds	Objective Function
$F_1$	2	$[0.1, 1] \cup [0, 5]$	Function CONSTR in Ref. [4]
$F_2$	2	$[0, \pi]^n$	Function TNK in Ref. [4]
$F_3$	2	$[0.1, 1]^n$	Test Function 4 in Ref. [5]
$UF_1$	30	$[0, 1] \times [-1, 1]^{n-1}$	Refer to Ref. [36]
$UF_2$	30	$[0, 1] \times [-1, 1]^{n-1}$	Refer to Ref. [36]
$UF_3$	30	$[0, 1]^n$	Refer to Ref. [36]
$ZDT_1$	30	$[0, 1]^n$	Refer to Ref. [6]
$ZDT_3$	30	$[0, 1]^n$	Refer to Ref. [6]
$DTLZ_1$	10	$[0, 1]^n$	Refer to Ref. [35]
$DTLZ_2$	10	$[0, 1]^n$	Refer to Ref. [35]
$DTLZ_4$	10	$[0, 1]^n$	Refer to Ref. [35]
$DTLZ_5$	10	$[0, 1]^n$	Refer to Ref. [35]

Table 3. The simulation parameters for compared methods

Methods	Parameter setting
NSGA-II	$P_c = 0.9, P_m = 0.1$
BMOPSO	$JP = 0.001, \alpha = 6$
MOEA/D	$F = 0.5, P_c = 0.5$
TV-MOPSO	$w_{max} = 0.7, w_{min} = 0.4, c_{1s} = c_{2f} = 2.5, c_{1f} = c_{2s} = 0.5$

### 4.2.2. Simulation results on 12 benchmark test functions

The statistical results of the Monte-Carlo experiments with respect to the five performance metrics are reported in Tables 4-8. In these tables, the best average results obtained with regarding to each metric are highlighted in **boldface**. Note that the Pareto fronts obtained by different MOO algorithms are depicted in **Appendix**.

Table 4. Statistical results of NNS obtained by different methods for different test functions

Functions	NNS	Methods				
		SAMOPSO	MOEA/D	BMOPSO	TV-MOPSO	NSGA-II
$F_1$	Best	6.50E+01	5.40E+01	3.60E+01	1.50E+01	1.90E+01
	Average	<b>5.38E+01</b>	4.68E+01	2.39E+01	1.23E+01	7.80E+00
	Worst	4.40E+01	3.90E+01	1.60E+01	8.00E+00	5.00E+00
	Std. Dev.	5.35E+00	6.37E+00	6.52E+00	2.45E+00	4.26E+00
$F_2$	Best	2.80E+01	2.00E+01	2.20E+01	2.80E+01	2.10E+01
	Average	<b>2.00E+01</b>	1.74E+01	1.70E+01	1.83E+01	1.63E+01
	Worst	1.40E+01	1.20E+01	1.10E+01	1.30E+01	1.10E+01
	Std. Dev.	4.19E+00	2.27E+00	3.83E+00	5.10E+00	3.46E+00
$F_3$	Best	9.90E+01	9.30E+01	7.70E+01	5.60E+01	5.50E+01
	Average	<b>9.56E+01</b>	3.75E+01	4.78E+01	3.10E+01	2.20E+01
	Worst	7.90E+01	1.10E+01	1.70E+01	9.00E+00	3.00E+00
	Std. Dev.	6.11E+00	3.43E+01	2.38E+01	1.70E+01	2.17E+01
$UF_1$	Best	3.50E+01	4.00E+00	4.00E+00	3.20E+01	3.00E+00
	Average	<b>2.67E+01</b>	1.50E+00	1.00E+00	1.95E+01	8.00E-01
	Worst	2.00E+01	0.00E+00	0.00E+00	2.00E+00	0.00E+00
	Std. Dev.	4.67E+00	1.51E+00	1.15E+00	8.96E+00	1.32E+00
$UF_2$	Best	4.90E+01	1.40E+01	1.80E+01	4.00E+00	1.00E+00
	Average	<b>2.53E+01</b>	3.30E+00	3.80E+00	7.00E-01	3.00E-01
	Worst	9.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	1.25E+01	6.20E+00	5.33E+00	1.34E+00	4.83E-01
$UF_3$	Best	6.90E+01	3.20E+01	3.00E+00	5.00E+00	0.00E+00
	Average	<b>2.35E+01</b>	6.70E+00	5.00E-01	1.00E+00	0.00E+00
	Worst	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	2.16E+01	9.82E+00	2.26E-01	3.16E-01	0.00E+00
$ZDT1$	Best	9.00E+01	2.90E+01	8.80E+01	2.00E+01	1.60E+01
	Average	<b>7.35E+01</b>	1.22E+01	4.08E+01	9.60E+00	7.80E+00
	Worst	4.30E+01	5.00E+00	1.10E+01	2.00E+00	3.00E+00
	Std. Dev.	1.38E+01	6.75E+00	3.20E+01	6.28E+00	4.32E+00
$ZDT_3$	Best	8.90E+01	1.90E+01	6.30E+01	1.80E+01	7.00E+00
	Average	<b>5.60E+01</b>	1.34E+01	1.35E+01	9.90E+00	3.40E+00
	Worst	4.30E+01	8.00E+00	9.00E+00	1.00E+00	1.00E+00
	Std. Dev.	1.37E+01	4.25E+00	1.94E+01	5.86E+00	2.07E+00
$DTLZ_1$	Best	3.00E+00	1.60E+01	1.00E+00	0.00E+00	0.00E+00
	Average	7.00E-01	<b>3.90E+00</b>	1.00E-01	0.00E+00	0.00E+00
	Worst	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	1.06E+00	4.89E+00	3.16E-01	0.00E+00	0.00E+00
$DTLZ_2$	Best	5.50E+01	5.70E+01	4.20E+01	1.80E+01	1.70E+01
	Average	<b>3.80E+01</b>	3.56E+01	2.72E+01	1.41E+01	1.26E+01
	Worst	2.50E+01	1.10E+01	1.60E+01	1.00E+01	0.00E+00
	Std. Dev.	7.43E+00	8.68E+00	8.11E+00	2.85E+00	8.88E+00
$DTLZ_4$	Best	8.90E+01	3.70E+01	5.70E+01	2.60E+01	5.00E+00
	Average	<b>2.21E+01</b>	1.44E+01	1.88E+01	9.60E+00	1.30E+00
	Worst	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std. Dev.	2.76E+01	1.25E+01	1.63E+01	1.06E+01	1.83E+00
$DTLZ_5$	Best	8.50E+01	1.80E+01	8.20E+01	1.50E+01	7.00E+00
	Average	<b>3.64E+01</b>	7.20E+00	3.63E+01	6.10E+00	1.70E+00
	Worst	2.30E+01	0.00E+00	1.20E+01	0.00E+00	0.00E+00
	Std. Dev.	1.71E+01	6.80E+00	3.83E+01	4.77E+00	2.36E+00

Table 5. Statistical results of  $ER$  obtained by different methods for different test functions

Functions	ER	Methods				
		SAMOPSO	MOEA/D	BMOPSO	TV-MOPSO	NSGA-II
$F_1$	Best	3.50E-01	4.60E-01	6.40E-01	8.50E-01	8.10E-01
	Average	<b>4.62E-01</b>	5.32E-01	7.61E-01	8.77E-01	9.22E-01
	Worst	5.60E-01	6.10E-01	8.40E-01	9.20E-01	9.50E-01
	Std. Dev.	5.35E-02	6.37E-02	6.53E-02	2.46E-02	4.26E-02
$F_2$	Best	7.20E-01	8.00E-01	7.80E-01	7.20E-01	7.90E-01
	Average	<b>8.00E-01</b>	8.26E-01	8.30E-01	8.17E-01	8.37E-02
	Worst	8.60E-01	8.80E-01	8.90E-01	8.70E-01	8.90E-01
	Std. Dev.	4.19E-02	2.27E-02	3.83E-02	5.09E-02	3.46E-02
$F_3$	Best	1.00E-02	7.00E-02	2.30E-01	4.40E-01	4.50E-01
	Average	<b>4.40E-02</b>	6.25E-01	5.22E-01	6.90E-01	7.80E-01
	Worst	2.10E-01	8.90E-01	8.30E-01	9.10E-01	9.70E-01
	Std. Dev.	6.11E-02	3.43E-01	2.37E-01	1.71E-01	2.17E-01
$UF_1$	Best	6.50E-01	9.60E-01	9.60E-01	6.80E-01	9.70E-01
	Average	<b>7.33E-01</b>	9.85E-01	9.90E-01	8.05E-01	9.92E-01
	Worst	8.0E-01	1.00E+00	1.00E+00	9.80E-01	1.00E+00
	Std. Dev.	4.67E-02	1.51E-02	1.15E-02	8.96E-02	1.32E-02
$UF_2$	Best	5.10E-01	8.60E-01	8.20E-01	9.60E-01	9.90E-01
	Average	<b>7.47E-01</b>	9.67E-01	9.62E-01	9.93E-01	9.97E-01
	Worst	9.10E-01	1.00E+00	1.00E+00	1.00E+00	1.00E+00
	Std. Dev.	1.25E-01	6.20E-02	5.32E-02	1.34E-02	4.83E-03
$UF_3$	Best	3.10E-01	6.80E-01	9.70E-02	9.50E-01	1.00E+00
	Average	<b>7.65E-01</b>	9.33E-01	9.95E-02	9.90E-01	1.00E+00
	Worst	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00
	Std. Dev.	2.16E-01	9.82E-02	2.26E-03	3.16E-03	0.00E+00
$ZDT1$	Best	1.00E-01	7.10E-01	1.20E-01	8.00E-01	8.40E-01
	Average	<b>2.65E-01</b>	8.78E-01	5.92E-01	9.04E-01	9.22E-01
	Worst	5.70E-01	9.50E-01	8.90E-01	9.80E-01	9.70E-01
	Std. Dev.	1.38E-01	6.75E-02	3.20E-01	6.28E-02	4.32E-02
$ZDT3$	Best	1.10E-01	8.10E-01	3.70E-01	8.20E-01	9.30E-01
	Average	<b>4.40E-01</b>	8.66E-01	8.65E-01	9.01E-01	9.66E-01
	Worst	5.70E-01	9.20E-01	9.10E-01	9.90E-01	9.90E-01
	Std. Dev.	1.37E-01	4.25E-02	1.94E-01	5.86E-02	2.07E-02
$DTLZ_1$	Best	9.70E-01	8.40E-01	9.90E-01	1.00E+00	1.00E+00
	Average	9.93E-01	<b>9.61E-01</b>	9.99E-01	1.00E+00	1.00E+00
	Worst	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00
	Std. Dev.	1.06E-02	4.89E-02	3.16E-03	0.00E+00	0.00E+00
$DTLZ_2$	Best	4.50E-01	4.30E-01	5.80E-01	8.20E-01	8.30E-01
	Average	<b>6.20E-01</b>	6.44E-01	7.28E-01	8.59E-01	8.74E-01
	Worst	7.50E-01	8.90E-01	8.40E-01	9.00E-01	1.00E+00
	Std. Dev.	7.43E-02	8.68E-02	8.11E-02	2.85E-02	8.88E-02
$DTLZ_4$	Best	1.10E-01	6.30E-01	4.30E-01	7.40E-01	9.50E-01
	Average	<b>7.79E-01</b>	8.56E-01	8.12E-01	9.04E-01	9.87E-01
	Worst	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00
	Std. Dev.	2.76E-01	1.25E-01	1.63E-01	1.06E-01	1.83E-02
$DTLZ_5$	Best	1.50E-01	8.20E-01	1.80E-01	8.50E-01	9.30E-01
	Average	<b>6.36E-01</b>	9.28E-01	6.37E-01	9.39E-01	9.83E-01
	Worst	7.70E-01	1.00E+00	8.80E-01	1.00E+00	1.00E+00
	Std. Dev.	1.71E-01	6.80E-02	3.83E-01	4.77E-02	2.36E-02



Table 6. Statistical results of  $GD$  obtained by different methods for different test functions

Functions	GD	Methods				
		SAMOPSO	MOEA/D	BMOPSO	TV-MOPSO	NSGA-II
$F_1$	Best	4.90E-04	5.71E-04	6.79E-04	1.07E-03	1.34E-03
	Average	<b>6.29E-04</b>	7.65E-04	1.75E-03	2.39E-03	2.49E-03
	Worst	8.41E-04	9.51E-04	2.70E-03	5.30E-03	4.67E-03
	Std. Dev.	1.02E-04	1.12E-04	6.33E-04	1.34E-03	1.02E-03
$F_2$	Best	2.72E-04	2.63E-04	2.98E-04	2.75E-04	2.98E-04
	Average	<b>3.24E-04</b>	3.59E-04	3.66E-04	3.58E-04	3.70E-04
	Worst	4.01E-04	4.62E-04	4.64E-04	4.60E-04	6.82E-04
	Std. Dev.	3.94E-05	5.28E-05	5.97E-05	5.79E-05	1.14E-04
$F_3$	Best	3.90E-07	9.85E-05	6.17E-05	7.04E-05	2.69E-04
	Average	<b>3.17E-06</b>	1.69E-02	2.42E-04	2.71E-02	3.07E-02
	Worst	1.99E-05	1.20E-01	4.96E-04	1.39E-01	1.41E-01
	Std. Dev.	6.39E-05	3.73E-02	1.58E-04	5.67E-02	5.29E-02
$UF_1$	Best	5.90E-04	3.14E-03	5.74E-03	9.13E-04	5.81E-03
	Average	<b>1.33E-03</b>	1.55E-02	2.91E-02	2.67E-03	3.32E-02
	Worst	3.06E-03	3.83E-02	7.34E-02	9.20E-03	5.01E-02
	Std. Dev.	9.31E-04	1.45E-02	2.42E-02	2.46E-03	1.54E-02
$UF_2$	Best	7.57E-04	1.79E-03	1.67E-03	2.77E-03	2.45E-03
	Average	<b>1.07E-03</b>	2.29E-03	2.20E-03	3.18E-03	4.13E-03
	Worst	1.85E-03	2.76E-03	2.96E-03	3.66E-03	8.21E-03
	Std. Dev.	3.21E-04	2.29E-03	4.46E-04	2.89E-04	1.72E-03
$UF_3$	Best	3.81E-04	5.28E-04	6.84E-03	4.27E-03	5.56E-03
	Average	<b>1.63E-03</b>	6.75E-03	1.63E-02	8.85E-03	1.69E-02
	Worst	9.44E-03	2.72E-02	2.82E-02	1.43E-02	4.29E-02
	Std. Dev.	2.78E-03	8.66E-03	6.88E-03	3.78E-03	1.12E-02
$ZDT_1$	Best	1.56E-06	1.77E-04	2.01E-06	1.40E-04	1.68E-03
	Average	<b>5.29E-05</b>	2.88E-04	2.43E-04	3.24E-04	2.99E-03
	Worst	1.13E-04	4.22E-04	1.10E-03	6.34E-04	5.16E-03
	Std. Dev.	4.20E-05	6.37E-05	3.41E-04	1.35E-04	1.17E-03
$ZDT_3$	Best	2.89E-05	2.77E-04	2.75E-04	3.72E-05	3.01E-03
	Average	<b>8.94E-05</b>	4.16E-04	3.87E-04	5.27E-04	7.96E-03
	Worst	2.03E-04	6.01E-04	4.87E-04	1.82E-03	2.22E-02
	Std. Dev.	4.99E-05	8.79E-05	6.87E-05	5.42E-04	5.86E-03
$DTLZ_1$	Best	2.32E+00	2.18E-01	3.32E+00	3.68E+00	3.74E+00
	Average	3.16E+00	<b>1.72E+00</b>	3.43E+00	3.99E+00	4.62E+00
	Worst	3.86E+00	5.12E+00	5.21E+00	4.97E+00	5.70E+00
	Std. Dev.	4.57E-01	1.42E+00	1.27E+00	3.63E-01	5.99E-01
$DTLZ_2$	Best	9.97E-04	1.84E-03	2.48E-03	2.52E-03	6.31E-03
	Average	<b>2.17E-03</b>	2.72E-03	3.20E-03	4.21E-03	7.75E-03
	Worst	4.39E-03	3.60E-03	4.08E-03	6.25E-03	9.31E-03
	Std. Dev.	1.17E-03	4.86E-04	4.88E-04	1.16E-03	9.13E-04
$DTLZ_4$	Best	1.17E-03	3.46E-03	2.93E-03	3.63E-03	4.13E-03
	Average	<b>5.05E-03</b>	8.29E-03	7.95E-03	1.68E-02	2.79E-02
	Worst	1.77E-02	3.21E-02	1.25E-02	3.53E-02	4.61E-02
	Std. Dev.	5.13E-03	1.02E-02	2.64E-03	9.99E-03	1.36E-02
$DTLZ_5$	Best	8.08E-05	6.35E-04	1.46E-04	5.48E-04	1.16E-03
	Average	<b>2.81E-04</b>	1.36E-03	4.60E-04	1.68E-03	2.08E-03
	Worst	6.05E-04	1.73E-03	9.41E-04	2.59E-03	3.77E-03
	Std. Dev.	2.07E-04	3.24E-04	2.34E-04	5.84E-04	9.13E-04

Table 7. Statistical results of  $SM$  obtained by different methods for different test functions

Functions	SM	Methods				
		SAMOPSO	MOEA/D	BMOPSO	TV-MOPSO	NSGA-II
$F_1$	Best	4.11E-03	4.61E-03	4.15E-03	3.72E-03	3.88E-03
	Average	<b>5.23E-03</b>	6.02E-03	7.23E-03	7.79E-03	8.97E-03
	Worst	6.75E-03	7.35E-03	1.56E-02	1.79E-02	1.44E-02
	Std. Dev.	7.95E-04	7.84E-04	3.59E-03	4.52E-03	3.20E-03
$F_2$	Best	1.86E-03	1.88E-03	2.04E-03	1.86E-03	2.22E-03
	Average	<b>2.39E-03</b>	2.63E-03	2.66E-03	2.59E-03	2.74E-03
	Worst	3.01E-03	3.43E-03	3.73E-03	3.42E-03	3.77E-03
	Std. Dev.	3.52E-04	5.67E-04	7.10E-04	5.09E-04	4.96E-04
$F_3$	Best	3.82E-06	8.89E-04	5.87E-04	6.97E-05	2.08E-03
	Average	<b>3.01E-05</b>	1.23E-01	1.94E-03	2.34E-01	2.76E-01
	Worst	1.84E-04	1.04E+00	3.71E-03	1.19E+00	1.26E+00
	Std. Dev.	5.96E-05	3.23E-01	1.13E-03	4.90E-01	4.68E-01
$UF_1$	Best	4.45E-03	1.52E-02	1.90E-02	6.43E-03	2.02E-02
	Average	<b>1.16E-02</b>	9.30E-02	1.56E-01	2.14E-02	2.13E-01
	Worst	2.93E-02	2.39E-01	3.78E-01	6.28E-02	3.16E-01
	Std. Dev.	9.05E-03	9.64E-02	1.36E-01	1.67E-02	1.03E-01
$UF_2$	Best	6.33E-03	8.94E-03	9.92E-03	1.02E-02	2.21E-02
	Average	<b>8.69E-03</b>	1.53E-02	1.47E-02	1.96E-02	2.61E-02
	Worst	1.69E-02	2.04E-02	1.99E-02	3.18E-02	3.19E-02
	Std. Dev.	3.15E-03	3.42E-03	3.58E-03	6.22E-03	3.09E-03
$UF_3$	Best	3.14E-03	1.22E-02	1.61E-02	3.74E-03	3.36E-02
	Average	<b>1.19E-02</b>	3.27E-02	7.60E-02	4.38E-02	1.01E-01
	Worst	7.29E-02	6.23E-02	2.49E-01	1.72E-01	1.41E-01
	Std. Dev.	2.16E-02	2.00E-02	7.61E-02	5.62E-02	3.84E-02
$ZDT_1$	Best	1.49E-05	1.03E-03	1.93E-05	8.91E-04	7.18E-03
	Average	<b>4.74E-04</b>	1.73E-03	1.03E-03	1.88E-03	1.23E-02
	Worst	1.07E-03	2.62E-03	2.69E-03	3.31E-03	2.33E-02
	Std. Dev.	3.72E-04	4.47E-04	9.83E-04	6.80E-04	5.01E-03
$ZDT_3$	Best	2.78E-04	1.83E-03	1.49E-03	3.17E-04	1.01E-02
	Average	<b>7.59E-04</b>	2.51E-03	2.48E-03	3.51E-03	2.55E-02
	Worst	1.91E-03	3.36E-03	3.33E-03	1.27E-02	4.67E-02
	Std. Dev.	4.68E-04	4.84E-04	4.74E-04	3.70E-03	1.30E-02
$DTLZ_1$	Best	8.20E-01	2.60E+00	4.67E+00	6.43E+00	7.57E+00
	Average	4.22E+00	<b>4.03E+00</b>	5.62E+00	6.56E+00	1.05E+01
	Worst	8.17E+00	4.70E+00	7.27E+00	2.40E+01	1.48E+01
	Std. Dev.	1.88E+00	6.08E-01	8.82E-01	6.97E+00	2.06E+00
$DTLZ_2$	Best	8.59E-03	1.46E-02	1.98E-02	1.81E-02	3.85E-02
	Average	<b>1.82E-02</b>	1.95E-02	2.32E-02	2.57E-02	5.17E-02
	Worst	4.10E-02	2.34E-02	2.85E-02	3.35E-02	6.61E-02
	Std. Dev.	1.17E-02	2.79E-03	2.96E-03	5.15E-03	7.77E-03
$DTLZ_4$	Best	8.81E-03	2.20E-02	3.40E-03	2.53E-02	3.59E-02
	Average	<b>2.75E-02</b>	5.81E-02	3.80E-02	6.95E-02	2.44E-01
	Worst	7.45E-02	9.99E-02	1.11E-01	1.03E-01	3.98E-01
	Std. Dev.	2.02E-02	2.88E-02	3.46E-02	2.26E-02	1.20E-01
$DTLZ_5$	Best	7.79E-04	4.26E-03	1.37E-03	5.16E-03	6.51E-03
	Average	<b>1.05E-03</b>	9.08E-03	3.79E-03	9.88E-03	1.12E-02
	Worst	1.43E-03	1.08E-02	8.83E-03	1.65E-02	1.95E-02
	Std. Dev.	2.22E-04	1.95E-03	2.14E-03	3.31E-03	3.83E-03

Table 8. Statistical results of *CT* (seconds) obtained by different methods for different test functions

Functions	CT	Methods				
		SAMOPSO	MOEA/D	BMOPSO	TV-MOPSO	NSGA-II
$F_1$	Best	8.14E+00	8.06E+00	1.58E+01	7.47E+00	9.16E+00
	Average	1.21E+01	1.18E+01	1.82E+01	<b>1.09E+01</b>	1.24E+01
	Worst	1.56E+01	1.52E+01	2.21E+01	1.32E+01	1.60E+01
	Std. Dev.	1.05E+00	1.40E+00	1.97E+00	1.16E+00	1.31E+00
$F_2$	Best	5.88E+00	5.66E+00	6.23E+00	5.15E+00	5.99E+00
	Average	8.08E+00	7.99E+00	9.53E+00	<b>7.72E+00</b>	8.23E+00
	Worst	9.72E+00	1.06E+01	1.33E+01	9.21E+00	1.11E+01
	Std. Dev.	1.33E+00	1.47E+00	1.83E+00	1.42E+00	1.78E+00
$F_3$	Best	6.78E+00	6.26E+00	7.57E+00	5.45E+00	6.90E+00
	Average	8.53E+00	8.78E+00	1.08E+01	<b>7.88E+00</b>	9.88E+00
	Worst	1.24E+01	1.02E+01	1.63E+01	9.91E+00	1.51E+01
	Std. Dev.	1.01E+00	1.17E+00	1.95E+00	1.51E+00	1.85E+00
$UF_1$	Best	3.04E+01	2.91E+01	3.57E+01	2.84E+01	3.31E+01
	Average	3.68E+01	3.43E+01	4.23E+01	<b>3.27E+01</b>	3.98E+01
	Worst	3.96E+01	3.75E+01	4.48E+01	3.51E+01	4.23E+01
	Std. Dev.	2.09E+00	2.31E+00	1.82E+00	2.35E+00	2.01E+00
$UF_2$	Best	3.12E+01	3.01E+01	3.76E+01	2.78E+01	3.50E+01
	Average	3.59E+01	3.31E+01	4.31E+01	<b>3.31E+01</b>	4.25E+01
	Worst	3.98E+01	3.79E+01	4.68E+01	3.65E+01	4.41E+01
	Std. Dev.	2.22E+00	3.01E+00	3.57E+00	2.71E+00	3.05E+00
$UF_3$	Best	3.24E+01	3.22E+01	3.98E+01	2.95E+01	3.74E+01
	Average	3.79E+01	3.52E+01	4.52E+01	<b>3.52E+01</b>	4.45E+01
	Worst	4.12E+01	3.98E+01	4.83E+01	3.83E+01	4.63E+01
	Std. Dev.	2.51E+00	3.17E+00	3.68E+00	2.89E+00	3.11E+00
$ZDT_1$	Best	2.92E+01	3.01E+01	3.52E+01	2.58E+01	3.37E+01
	Average	3.41E+01	3.31E+01	4.13E+01	<b>3.16E+01</b>	4.09E+01
	Worst	3.73E+01	3.79E+01	4.46E+01	3.41E+01	4.20E+01
	Std. Dev.	2.05E+00	3.01E+00	2.81E+00	2.52E+00	2.99E+00
$ZDT_3$	Best	2.81E+01	2.91E+01	3.30E+01	2.48E+01	3.26E+01
	Average	3.37E+01	3.28E+01	39.8E+01	<b>2.97E+01</b>	38.5E+01
	Worst	3.63E+01	3.64E+01	4.26E+01	3.36E+01	4.17E+01
	Std. Dev.	2.02E+00	2.99E+00	2.41E+00	2.18E+00	2.76E+00
$DTLZ_1$	Best	1.62E+01	1.74E+01	2.13E+01	1.31E+01	2.03E+01
	Average	2.17E+01	2.07E+01	27.6E+01	<b>1.79E+01</b>	26.6E+01
	Worst	2.41E+01	2.54E+01	3.06E+01	2.13E+01	3.27E+01
	Std. Dev.	2.35E+00	2.61E+00	1.91E+00	1.98E+00	3.01E+00
$DTLZ_2$	Best	1.53E+01	1.59E+01	2.01E+01	1.23E+01	1.94E+01
	Average	1.98E+01	1.95E+01	2.62E+01	<b>1.59E+01</b>	2.58E+01
	Worst	2.29E+01	2.42E+01	2.84E+01	2.00E+01	2.97E+01
	Std. Dev.	1.75E+00	1.83E+00	1.51E+00	1.83E+00	1.94E+00
$DTLZ_4$	Best	1.93E+01	2.02E+01	2.54E+01	1.73E+01	2.53E+01
	Average	2.35E+01	2.23E+01	2.92E+01	<b>1.92E+01</b>	2.86E+01
	Worst	2.69E+01	3.22E+01	3.44E+01	2.60E+01	3.77E+01
	Std. Dev.	1.93E+00	2.03E+00	1.99E+00	2.07E+00	2.14E+00
$DTLZ_5$	Best	1.64E+01	1.70E+01	2.16E+01	1.43E+01	2.04E+01
	Average	2.05E+01	1.92E+01	2.72E+01	<b>1.61E+01</b>	2.65E+01
	Worst	2.31E+01	2.62E+01	2.94E+01	2.09E+01	3.17E+01
	Std. Dev.	1.65E+00	1.53E+00	1.65E+00	1.78E+00	1.89E+00

#### 4.2.3. Analysis

From the statistical results of *NNS*, *ER*, *GD* and *SM* reported in Tables 4-7, it can be observed that, our proposed method generally outperforms the four MOO algorithms compared over the majority of the benchmarks (except Function  $DTLT_1$ ). As shown in Tables 4-7, we can observe that all MOO methods cannot efficiently solve Function  $DTLT_1$ . This could be interpreted by the complexities of objectives and many local optimal solutions contained in the Pareto front of Function  $DTLT_1$  [35]. Nevertheless, MOEA/D and SAMOPSO outperforms the other three over this test function in terms of the average values of the *NNS*, *ER*, *GD* and *SM* metrics.

From Table 8, it is evident that TV-MOPSO outperforms MOEA/D, SAMOPSO, NSGA-II and BMOPSO with respect to the average computation time. Since TV-MOPSO has the simplest updating rules in adjusting the three control parameters of particles, its computation time is likely reduced. However, it is important to note that, despite consuming slightly more computation time than MOEA/D and TV-MOPSO, the proposed method provides the best performance in terms of the quality of Pareto front in most of the benchmark test functions as shown in Tables 4-7. Also, the difference in the average computation time between SAMOPSO, MOEA/D and TV-MOPSO is very small in all benchmark test functions. This implies that the computation time of SAMOPSO is comparable with those of

other MOO algorithms tested.

#### 4.2.4. Statistical Comparison

This subsection performs a statistical comparison to detect whether the five tested methods are significantly different in solving the 12 benchmarks. In the statistical comparison, a rank-based analysis is first conducted to examine the average rank of each method over the 12 benchmark functions. Then, the non-parametric Friedman test [37] on the mean rank followed by the pairwise post hoc Bonferroni-Dunn test [37] is performed to investigate the overall performances of different methods over the 12 test functions. Note that only the *NNS* metric is considered as an example in the statistical comparison. Following the same statistical comparison for the remaining three evaluation metrics, i.e. *ER*, *GD* and *SM*, we can readily analyse how much the five methods are significantly different from each other.

The ranks and average rank of each method for all benchmark functions are summarised in Table 9. From this table, SAMOPSO outperforms MOEA/D, BMOPSO, TV-MOPSO and NSGA-II with respect to the average *NNS* performance. Because this study compares 5 methods on 12 benchmark cases, the F-statistic value of the Friedman test at the confidence level of 90% equals to 2.0772. Note that the F-statistic value of the Friedman test at the confidence level of  $\alpha\%$  can be gained using the Matlab command:  $\text{finv}(\alpha, K-1, (K-1)(N-1))$ , where  $K$  and  $N$ , respectively, denote the number of methods and test functions. From Table 9, the obtained Friedman statistic value ( $F_{score}$ ) is equal to 31.6954. For more details about the calculation of  $F_{score}$ , the reader is referred to [37]. Since the  $F_{score}$  value is bigger than the F-statistic value, the null hypothesis, i.e. each method equally performs over all considered benchmark problems, can be rejected [37]. This means that the five methods tested are significantly different over the 12 test functions at the confidence level of 90%.

Although the non-parametric Friedman test confirms that the five methods are significantly different over the 12 benchmark functions at the confidence level of 90%, it cannot be sufficiently concluded that SAMOPSO performs significantly better than the other four methods in terms of the *NNS* metric. To highlight the mean performance of SAMOPSO with respect to the other four methods, the pairwise post hoc Bonferroni-Dunn test is performed in this paper. To examine whether or not a given method is significantly better than another method at the confidence level of  $\alpha\%$ , the Bonferroni-Dunn test checks whether the average rank difference between the two methods is greater than the critical difference value (*CD*). If it is true, we can conclude that the given method is significantly better than the other method at the confidence level of  $\alpha\%$  [37]. Note that the *CD* value can be calculated by  $q_\alpha \sqrt{K(K+1)/(6N)}$  for  $K$  methods over  $N$  benchmark test functions [37]. Here,  $q_\alpha$  is a constant parameter and is equal to 2.241 in this study according to [37].

For the case where 5 methods are used to solve 12 test functions, we can readily obtain that, at the confidence level of 90%, the critical difference value *CD* of the Bonferroni-Dunn test equals to 1.4466. From Table 9, it can be easily evaluated that the average *NNS* value differences of SAMOPSO with respect to those of MOEA/D, BMOPSO, TV-MOPSO and NSGA-II are 1.5, 1.67, 2.5 and 3.83, which are all larger than the critical difference value of 1.4466. This implies that the proposed SAMOPSO provides significantly better average *NNS* performance than the four compared methods over the 12 chosen benchmark problems at the confidence level of 90%. Following the same analysis, we can also conclude that SAMOPSO performs significantly better than its peers in terms of the mean *ER*, *GD* and *SM* performance over the 12 benchmarks at the confidence level of 90%.

Table 9. Rank values of average *NNS* results of different methods for 12 benchmarks (“AVR.” denotes the average rank value)

	SAMOPSO	MOEA/D	BMOPSO	TV-MOPSO	NSGA-II
$F_1$	1	2	3	4	5
$F_2$	1	3	4	2	5
$F_3$	1	3	2	4	5
$UF_1$	1	3	4	2	5
$UF_2$	1	3	2	4	5
$UF_3$	1	2	4	3	5
$ZDT_1$	1	3	2	4	5
$ZDT_3$	1	3	2	4	5
$DTLZ_1$	2	1	3	4	4
$DTLZ_2$	1	2	3	4	5
$DTLZ_4$	1	3	2	4	5
$DTLZ_5$	1	3	2	4	5
AVR.	1.083	2.583	2.750	3.583	4.917

### 4.3. Application on the environmental/economic dispatch (EED) problem

#### 4.3.1. Formulation of EED problem

Aiming to determine the optimal combination of power outputs of all generators over the whole scheduling period, the EED problem can be mathematically formulated as follows [17]:

$$\text{minimise : } \begin{cases} F = \sum_{i=1}^N (a_i + b_i P_i + c_i P_i^2) \\ E = \sum_{i=1}^N (\alpha_i + \beta_i P_i + \gamma_i P_i^2 + \xi_i \exp(\lambda_i P_i)) \end{cases} \quad (47)$$

$$\text{Subject to : } \begin{cases} P_{i,min} \leq P_i \leq P_{i,max} \end{cases} \quad (48)$$

$$\begin{cases} \sum_{i=1}^N P_i = P_L + P_D \end{cases} \quad (49)$$

$$\begin{cases} P_L = \sum_{i=1}^N \sum_{j=1}^N P_i B_{ij} P_j + \sum_{i=1}^N P_i B_{i0} + B_{00} \end{cases} \quad (50)$$

where  $F$  denotes the total fuel cost in \$/h.  $E$  represents the total emission rate in Kg/h.  $a_i$ ,  $b_i$  and  $c_i$  are fuel cost coefficients of generator  $i$ .  $\beta_i$ ,  $\gamma_i$ ,  $\xi_i$  and  $\lambda_i$  are emission coefficients of generator  $i$ .  $P_i$  is the power output of generator  $i$  in MW.  $N$  denotes the total number of generators.  $P_{i,min}$  and  $P_{i,max}$  denote the minimum and maximum output limits of generator  $i$ , respectively.  $P_L$  is the total power loss of the power system in MW.  $P_D$  is the total power demand of the system in MW.  $B_{ij}$ ,  $B_{i0}$  and  $B_{00}$  represent the transmission loss coefficients.

#### 4.3.2. Numerical simulation for the EED problem

SAMOPSO, MOEA/D, BMOPSO, TV-MOPSO and NSGA-II are applied to solve the IEEE-30-bus system [38] with 6 generators. In the numerical simulation, a Monte-Carlo experiment with 30 runs is conducted. The size of the repository of each method is bounded to be 50 in each run of the Monte-Carlo experiment. The total power demand of the system is given as  $P_D = 2.2$ . The minimum and maximum output boundaries for the 6 generators are set to be  $P_{min} = [0.05, 0.05, 0.05, 0.05, 0.05, 0.05]$  and  $P_{max} = [0.5, 0.6, 1.0, 1.2, 1.0, 0.6]$ , respectively. The other coefficients for the IEEE-30-bus system are given as follows:

$$B_{00} = 0.0014 \quad (51)$$

$$B_{i0} = (1e - 03) \cdot [0.010731 \ 1.7704 \ -4.0645 \ 3.8453 \ 1.3832 \ 5.5503] \quad (52)$$

$$B_{ij} = \begin{bmatrix} 0.0218 & 0.0107 & -0.00036 & -0.0011 & 0.00055 & 0.0033 \\ 0.0107 & 0.01704 & -0.0001 & -0.00179 & 0.00026 & 0.0028 \\ -0.0004 & -0.0002 & 0.02459 & -0.01328 & -0.0018 & -0.0079 \\ -0.0011 & -0.00197 & -0.01328 & 0.0265 & 0.0098 & 0.0045 \\ 0.00055 & 0.00026 & -0.0118 & 0.0098 & 0.0216 & -0.0001 \\ 0.0033 & 0.0028 & -0.00792 & 0.0045 & -0.00012 & 0.02978 \end{bmatrix} \quad (53)$$

Table 10. The fuel cost and emission coefficients of each generator of the IEEE-30-bus system ( $G_i$  denotes the  $i$ th generator)

	$G_1$	$G_2$	$G_3$	$G_4$	$G_5$	$G_6$
$a$	10	10	20	10	20	10
$b$	200	150	180	100	180	150
$c$	100	120	40	60	40	100
$\alpha$	4.091	2.543	4.258	5.526	4.258	6.131
$\beta$	-5.554	-6.047	-5.094	-3.55	-5.094	-5.555
$\gamma$	6.490	5.638	4.586	3.380	4.586	5.151
$\xi$	2e-04	5e-04	1e-06	2e-03	1e-06	1e-05
$\lambda$	2.857	3.333	8.000	2.000	8.000	6.667

The statistical results of the tested MOO algorithms, with respect to the five performance metrics, are summarised in Table 11. The non-dominated solutions produced by different methods are visualised in Fig. 5. From Table 11, it is clear that SAMOPSO outperforms other four algorithms in terms of the average  $NNS$ ,  $ER$ ,  $GD$  and  $SM$  performance in the EED problem. Comparing with MOEA/D, BMOPSO, TV-MOPSO and NSGA-II, SAMOPSO averagely improves the  $NNS$  performance by 97.87%, 153.64%, 298.57% and 654.05%; the  $ER$  performance by 38.44%, 43.33%, 48.60% and 52.27%; the  $GD$  performance by 50.71%, 53.99%, 55.41% and 55.57% and the  $SM$  performance by 49.76%, 54.22%, 57.26% and 57.96%, respectively. This confirms that SAMOPSO provides better quality of the Pareto front for the EED problem, compared with its competitors.

As shown in Table 11, TV-MOPSO, MOEA/D and SAMOPSO are ranked the first, second and third in terms of the computation time, but the difference is small. Here, it is important to note that, despite taking slightly more computation time than MOEA/D and TV-MOPSO, SAMOPSO significantly outperforms these two methods with respect to the mean  $NNS$ ,  $ER$ ,  $GD$  and  $SM$  performance.

Table 11. Statistical results of different evaluation metrics obtained by different methods for the EED problem

Metrics		Methods				
		SAMOPSO	MOEA/D	BMOPSO	TV-MOPSO	NSGA-II
$NNS$	Best	4.90e+01	2.90E+01	2.70E+01	1.70E+01	8.00E+00
	Average	<b>2.79E+01</b>	1.41E+01	1.10E+01	7.00E+00	3.70E+00
	Worst	6.00E+00	4.00E+00	4.00E+00	2.00E+00	1.00E+00
	Std. Dev.	1.83E+01	1.16E+01	8.69E+00	4.50E+00	2.00E+00
$ER$	Best	2.00E-02	4.20E-01	4.60E-01	6.60E-01	8.40E-01
	Average	<b>4.42E-01</b>	7.18E-01	7.80E-01	8.60E-01	9.26E-01
	Worst	8.80E-01	9.20E-01	9.20E-01	9.60E-01	9.80E-01
	Std. Dev.	3.66E-01	2.31E-01	1.74E-01	8.99E-02	4.01E-02
$GD$	Best	7.40E-04	1.44E-02	1.88E-02	1.21E-02	1.45E-02
	Average	<b>1.73E-02</b>	3.51E-02	3.76E-02	3.88E-02	3.91E-02
	Worst	3.59E-02	7.53E-02	7.96E-02	1.44E-01	1.21E-01
	Std. Dev.	1.31E-02	1.94E-02	2.16E-02	3.89E-02	3.33E-02
$SM$	Best	5.24E-03	7.54E-02	9.69E-02	7.97E-02	7.37E-02
	Average	<b>1.03E-01</b>	2.05E-01	2.24E-01	2.41E-01	2.45E-01
	Worst	2.47E-01	4.99E-01	5.42E-01	8.15E-01	9.94E-01
	Std. Dev.	8.05E-02	1.41E-01	1.61E-01	2.33E-01	2.76E-01
$CT$	Best	3.33E+01	3.20E+01	3.76E+01	3.03E+01	3.87E+01
	Average	3.72E+01	3.61E+01	3.82E+01	<b>3.41E+01</b>	4.15E+01
	Worst	4.01E+01	3.92E+01	4.44E+01	3.69E+01	4.52E+01
	Std. Dev.	1.15E+00	1.12E+00	1.35E+00	1.01E+00	1.59E+00

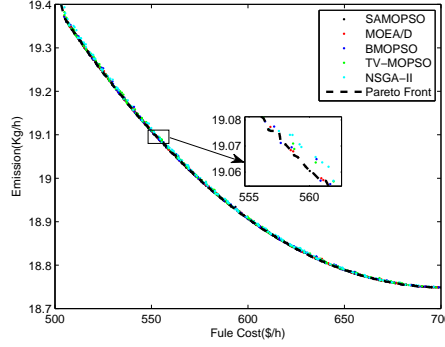


Fig. 5. Pareto fronts obtained by different methods for the EED problem

## 5. Conclusions

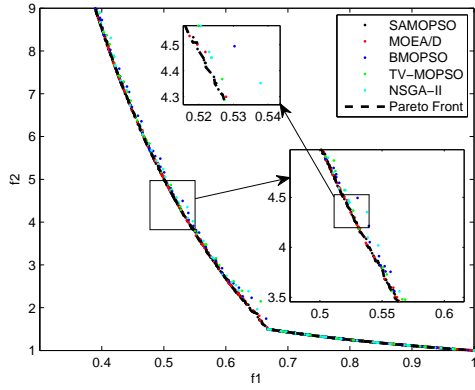
In this study, a PSO-based MOO framework is developed to obtain a high-quality Pareto front. For this purpose, a novel PSO method, named self-adaptive PSO (SAPSO), is first proposed and implemented to search the non-dominated solutions in the developed MOO framework. In order to well balance the trade-offs between the exploration and exploitation capabilities of SAPSO, we propose a self-adaptive strategy that tunes the three control parameters of particles. Since the convergence property of PSO plays a crucial role in the field of PSO development, this paper also investigates the convergence of the proposed SAPSO method with respect to different values of the three control parameters. Then, a convergence-guaranteed parameter selection principle is proposed for the developed SAPSO.

Leveraging the proposed SAPSO, this paper completes the design of a MOO framework, called SAMOPSO. In the proposed framework, a fixed-size external repository is designed to save non-dominated best solutions of particles. To obtain a well-distributed Pareto front, this paper develops the circular sorting method, which is integrated with the elitist-preserving approach [4] and updates the external repository.

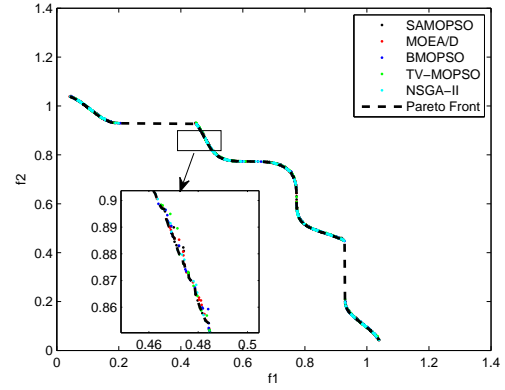
The proposed approach is validated via 12 benchmark test functions and a real-world MOO problem against four well-established MOO algorithms: MOEA/D, BMOPSO, NSGA-II and TV-MOPSO. The comparison is conducted based on five widely-adopted MOO performance metrics. The simulation results reveal that the proposed method is highly competitive in most of benchmark test functions with respect to the quality of the Pareto front. Moreover, the statistical analysis on the simulation results verifies that the proposed method significantly outperforms the other four compared algorithms in the selected benchmark problems at the confidence level of 90%. Also, the computation time of the proposed method is comparable with those of the other methods. This implies that the proposed framework is a very effective MOO algorithm.

## Appendix

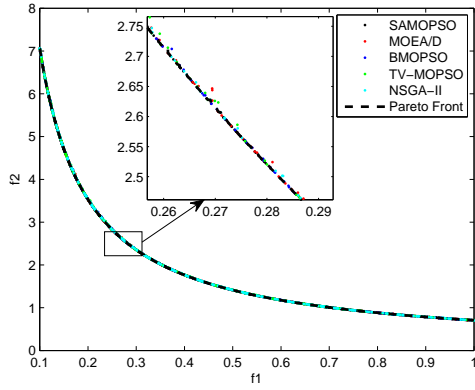
The obtained Pareto fronts of different methods for each benchmark test function are depicted in this Appendix.



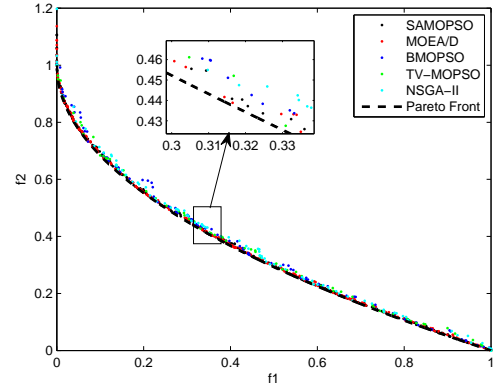
(a) Function  $F_1$



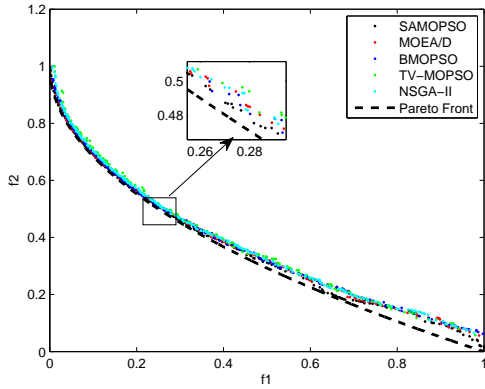
(b) Function  $F_2$



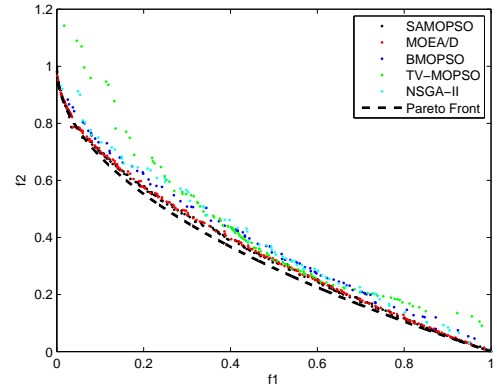
(c) Function  $F_3$



(d) Function  $UF_1$



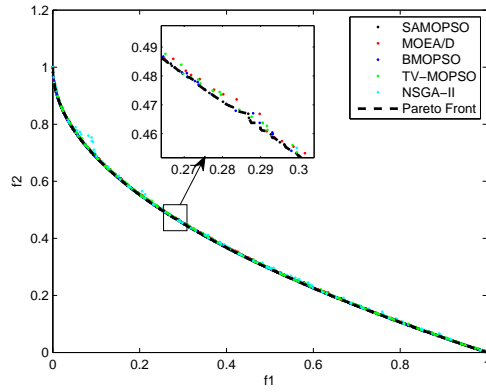
(e) Function  $UF_2$



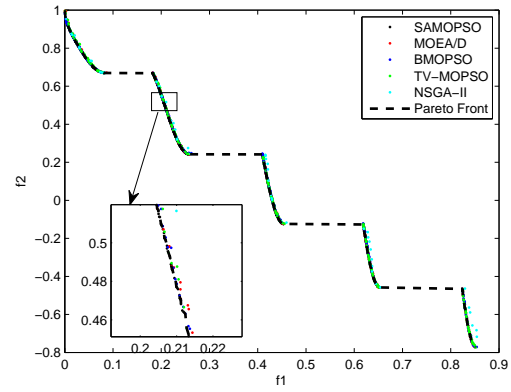
(f) Function  $UF_3$

Fig. 6. Pareto fronts obtained by different methods for different Functions

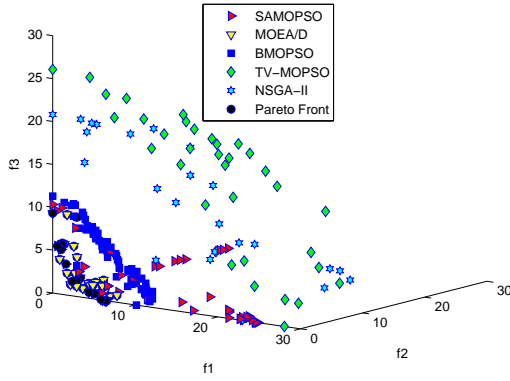




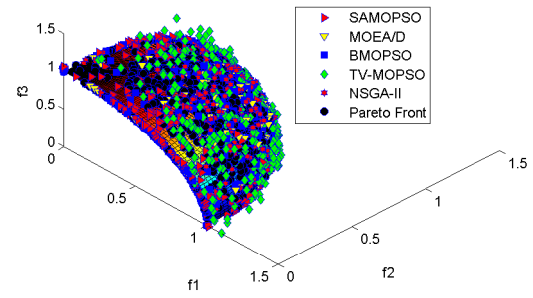
(a) Function  $ZDT_1$



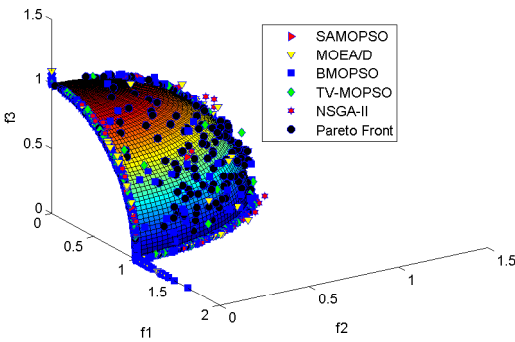
(b) Function  $ZDT_3$



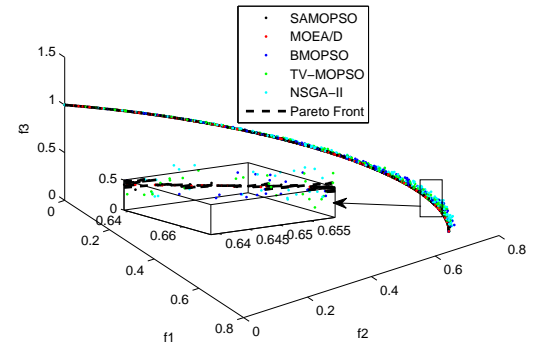
(c) Function  $DTLZ_1$



(d) Function  $DTLZ_2$



(e) Function  $DTLZ_4$



(f) Function  $DTLZ_5$

Fig. 7. Pareto fronts obtained by different methods for different Functions

## References

- [1] M. Arsuaga-Ríos, M. A. Vega-Rodríguez, F. Prieto-Castrillo, Meta-schedulers for grid computing based on multi-objective swarm algorithms, *Applied Soft Computing Journal* 13 (4) (2013) 1567–1582.

- [2] S. Nguyen, M. Zhang, M. Johnston, K. C. Tan, Automatic Design of Scheduling Policies for Dynamic Multi-objective Job Shop Scheduling via Cooperative Coevolution Genetic Programming, *IEEE Transactions on Evolutionary Computation* 18 (2) (2014) 193–208.
- [3] Y. Lu, J. Zhou, H. Qin, Y. Wang, Y. Zhang, Environmental/economic dispatch problem of power system by using an enhanced multi-objective differential evolution algorithm, *Energy Conversion and Management* 52 (2) (2011) 1175–1183.
- [4] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation* 6 (2) (2002) 182–197.
- [5] C. a. C. Coello, G. T. Pulido, M. S. Lechuga, Handling multiple objectives with particle swarm optimization, *IEEE Transactions on Evolutionary Computation* 8 (3) (2004) 256–279.
- [6] Y. Huo, Y. Zhuang, J. Gu, S. Ni, Elite-guided multi-objective artificial bee colony algorithm, *Applied Soft Computing* 32 (2015) 199–210.
- [7] Y. Wang, Y. Yang, Particle swarm optimization with preference order ranking for multi-objective optimization, *Information Sciences* 179 (12) (2009) 1944–1959.
- [8] D. Kundu, K. Suresh, S. Ghosh, S. Das, B. K. Panigrahi, S. Das, Multi-objective optimization with artificial weed colonies, *Information Sciences* 181 (12) (2011) 2441–2454.
- [9] S. M. A. Pahnkehloaei, A. Alfi, A. Sadollah, J. H. Kim, Gradient-based Water Cycle Algorithm with evaporation rate applied to chaos suppression, *Applied Soft Computing* 53 (2017) 420–440.
- [10] A. Arab, A. Alfi, An adaptive gradient descent-based local search in memetic algorithm applied to optimal controller design, *Information Sciences* 299 (2015) 117–142.
- [11] H. Li, Q. Zhang, Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II 13 (2) (2009) 184–302.
- [12] Y. Tan, Y. Jiao, H. Li, X. Wang, MOEA/D-SQA: a multi-objective memetic algorithm based on decomposition, *Engineering Optimization* 0273 (November 2012) (2012) 37–41.
- [13] S. Bandyopadhyay, R. Chakraborty, U. Maulik, Priority based dominance: A new measure in multiobjective optimization, *Information Sciences* 305 (2015) 97–109.
- [14] R. Wang, R. Purshouse, P. Fleming, Preference-Inspired Coevolutionary Algorithms for Many-Objective Optimization, *IEEE Transactions on Evolutionary Computation* 17 (4) (2013) 474–494.
- [15] A. Alfi, H. Modares, System identification and control using adaptive particle swarm optimization, *Applied Mathematical Modelling* 35 (3) (2011) 1210–1221.
- [16] Y. Mousavi, A. Alfi, A memetic algorithm applied to trajectory control by tuning of Fractional Order Proportional-Integral-Derivative controllers, *Applied Soft Computing Journal* 36 (2015) 599–617.
- [17] Y. Zhang, D.-W. Gong, Z. Ding, A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch, *Information Sciences* 192 (2012) 213–227.
- [18] K. Khalili-Damghani, A.-R. Abtahi, M. Tavana, A new multi-objective particle swarm optimization method for solving reliability redundancy allocation problems, *Reliability Engineering & System Safety* 111 (2013) 58–75.
- [19] S. V. Kamble, S. U. Mane, A. J. Umbarkar, Hybrid Multi-Objective Particle Swarm Optimization for Flexible Job Shop Scheduling Problem, *International Journal of Intelligent Systems and Applications* 7 (4) (2015) 54–61.
- [20] X. Shao, W. Liu, Q. Liu, C. Zhang, Hybrid discrete particle swarm optimization for multi-objective flexible job-shop scheduling problem, *The International Journal of Advanced Manufacturing Technology* 67 (9-12) (2013) 2885–2901.
- [21] P. K. Tripathi, S. Bandyopadhyay, S. K. Pal, Multi-Objective Particle Swarm Optimization with time variant inertia and acceleration coefficients, *Information Sciences* 177 (22) (2007) 5033–5049.
- [22] Y. Zhang, D.-w. Gong, J.-h. Zhang, Robot path planning in uncertain environment using multi-objective particle swarm optimization, *Neurocomputing* 103 (2013) 172–185.
- [23] J. H. Roh, M. J. Kim, H. Y. Song, J. B. Park, S. U. Lee, S. Y. Son, An improved mean-variance optimization for nonconvex economic dispatch problems, *Journal of Electrical Engineering and Technology* 8 (1) (2013) 80–89.
- [24] R. Akbari, K. Ziarati, A rank based particle swarm optimization algorithm with dynamic adaptation, *Journal of Computational and Applied Mathematics* 235 (8) (2011) 2694–2714.
- [25] A. Ratnaweera, S. K. Halgamuge, H. C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, *IEEE Transactions on Evolutionary Computation* 8 (3) (2004) 240–255.
- [26] A. Alfi, H. Modares, System identification and control using adaptive particle swarm optimization, *Applied Mathematical Modelling* 35 (3) (2011) 1210–1221.
- [27] C. a. Coello Coello, M. Reyes-Sierra, Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art, *International Journal of Computational Intelligence Research* 2 (3) (2006) 287–308.
- [28] P. Chakraborty, S. Das, G. G. Roy, A. Abraham, On convergence of the multi-objective particle swarm optimizers, *Information Sciences* 181 (8) (2011) 1411–1425.
- [29] C. Leboucher, H. S. Shin, P. Siarry, S. Le Ménéec, R. Chelouah, A. Tsourdos, Convergence proof of an enhanced Particle Swarm Optimisation method integrated with Evolutionary Game Theory, *Information Sciences* 346-347 (2016) 389–411.
- [30] Z. Yong, G. Dun-wei, Z. Wan-qiu, Feature selection of unreliable data using an improved multi-objective PSO algorithm, *Neurocomputing* 171 (2016) 1281–1290.
- [31] J. Zhang, Q. Tang, P. Li, D. Deng, Y. Chen, A modified MOEA/D approach to the solution of multi-objective optimal power flow problem, *Applied Soft Computing* 47 (2016) 494–514.
- [32] Y. Zhang, L. Wu, S. Wang, UCAV Path Planning by Fitness-Scaling Adaptive Chaotic Particle Swarm Optimization, *Mathematical Problems in Engineering* 2013.
- [33] Y. Zhang, X. Xiong, Q. Zhang, An Improved Self-Adaptive PSO Algorithm with Detection Function for Multimodal Function Optimization Problems, *Mathematical Problems in Engineering* (2013).
- [34] Z. Wang, S. Li, Z. Sang, A new constraint handling method based on the modified Alopex-based evolutionary algorithm, *Computers & Industrial Engineering* 73 (2014) 41–50.

- [35] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable multi-objective optimization test problems, Proceedings of the 2002 Congress on Evolutionary Computation, CEC 2002 1 (i) (2002) 825–830.
- [36] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, Multiobjective optimization Test Instances for the CEC 2009 Special Session and Competition, Technical Report (2009) 1–30.
- 530 [37] Demšar J, statistical comparisons of classifiers over multiple data sets, The Journal of Machine Learning Research 7 (2006) 1–30.
- [38] S. Hemamalini, S. P. Simon, Emission constrained economic dispatch with valve-point effect using particle swarm optimization, TENCON 2008 - 2008 IEEE Region 10 Conference (2008) 1–6.

2017-08-24

# A framework for multi-objective optimisation based on a new self-adaptive particle swarm optimisation algorithm

Tang, Biwei

Elsevier

---

Tang B, Zhua Z, Shin H-S, Tsourdos A, Luo J, A framework for multi-objective optimisation based on a new self-adaptive particle swarm optimisation algorithm, Information Services, Vol. 420, December 2017, pp. 364-385

<http://dx.doi.org/10.1016/j.ins.2017.08.076>

*Downloaded from Cranfield Library Services E-Repository*